

Learning Reach-to-Grasp Motions From Human Demonstrations

THÈSE N° 6434 (2014)

PRÉSENTÉE LE 28 NOVEMBRE 2014

À LA FACULTÉ DES SCIENCES ET TECHNIQUES DE L'INGÉNIEUR
LABORATOIRE D'ALGORITHMES ET SYSTÈMES D'APPRENTISSAGE
PROGRAMME DOCTORAL EN SYSTÈMES DE PRODUCTION ET ROBOTIQUE

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

POUR L'OBTENTION DU GRADE DE DOCTEUR ÈS SCIENCES

PAR

Ashwini SHUKLA

acceptée sur proposition du jury:

Dr A. Karimi, président du jury
Prof. A. Billard, directrice de thèse
Prof. R. Alterovitz, rapporteur
Prof. H. Bleuler, rapporteur
Prof. J. Steil, rapporteur



ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Suisse
2014

ABSTRACT

REACHING over to grasp an item is arguably the most commonly used motor skill by humans. Even under sudden perturbations, humans seem to react rapidly and adapt their motion to guarantee success. Despite the apparent ease and frequency with which we use this ability, a complete understanding of the underlying mechanisms cannot be claimed. It is partly due to such incomplete knowledge that adaptive robot motion for reaching and grasping under perturbations is not perfectly achieved. Approaches for modelling control schemes for achieving such tasks can be divided into two broad categories: a) The Generative approaches which model the complex underlying mechanisms and *explain* the output (here reach-to-grasp motions), and b) The Discriminative approaches which directly model the observed output and imitate with some generalization. In this thesis, we take the discriminative approach for modelling trajectories of reach-to-grasp motion from expert demonstrations. Throughout this thesis, we will employ time-independent (autonomous) flow based representations to learn reactive motion controllers which can then be ported onto robots.

This thesis is divided into three main parts. The first part is dedicated to biologically inspired modelling of reach-to-grasp motions with respect to the hand-arm coupling. We build upon previous work in motion modelling using autonomous dynamical systems (DS) and present a coupled dynamical system (CDS) model of these two subsystems. The coupled model ensures satisfaction of the constraints between the hand and the arm subsystems which are critical to the success of a reach-to-grasp task. Moreover, it reduces the complexity of the overall motion planning problem as compared to considering a combined problem for the hand and the arm motion.

In the second part we extend the CDS approach to incorporate multiple grasping points. Such a model is beneficial due to the fact that many daily life objects afford multiple grasping locations on their surface. The choice of a grasping point may be driven by the relative position/orientation of the object around the robot's end-effector and is critical under perturbations. If the target object is suddenly perturbed while motion is being executed toward a grasping point, the new pose of the object might render the original grasping point inaccessible. In such a scenario, it is desirable to switch to a different grasping point and adopt a new trajectory leading to it. We combine a DS based approach

with energy-function learning to learn a multiple attractor dynamical system where the attractors are mapped to the desired grasping points. We present the Augmented-SVM (ASVM) model that combines the classical SVM formulation with gradient constraints arising from the energy function to learn the desired dynamical function for motion generation.

We further explore the ASVM formulation as a more general function approximation technique. We show that it can be used for approximating an arbitrary function where a combination of equality or inequality constraints are specified on its value or gradient at a given set of inputs. On similar lines as the classical SVM, we formulate the ν parametrized version of the ASVM which explicitly provides a lower bound on the number of support vectors obtained.

In the last part of this thesis, we address the problem of inverse-kinematics and obstacle avoidance by combining our flow-based motion generator with global configuration-space planners. We claim that the two techniques complement each other. On one hand, the fast reactive nature of our flow based motion generator can be used to guide the search of a randomly exploring random tree (RRT) based global planner. On the other hand, global planners can efficiently handle arbitrary obstacles and avoid local minima present in the dynamical function learned from demonstrations. We show that combining the information from demonstrations with global planning in the form of a energy-map considerably decreases the computational complexity of state-of-the-art sampling based planners.

We have evaluated our models in simulations and real platforms of various robots, e.g., the 53 degree of freedom (DOF) humanoid robot iCub, 7 DOF KUKA LWR arm fitted with 4 DOF Barrett Hand and the 16 DOF Allegro Hand. We believe that this thesis has the following contributions to Robotics and Machine Learning. First, we have developed algorithms for fast and adaptive motion generation for reach-grasp motions. Second, we formulated an extension to the classical SVM formulation that takes into account the gradient information from data. We showed that instead of being limited as a classifier or a regressor, the SVM framework can be used as a more general function approximation technique. Lastly, we have combined our local methods with global approaches for planning to achieve arbitrary obstacle avoidance and considerable reduction in the computation complexity of the global planners.

Keywords: Reaching Movements, Hand arm coupling, Nonlinear dynamical system, Coupled dynamical system, Multiple attractor dynamical system, Imitation learning, Support vector machine, Gradient constraints.

RÉSUMÉ

ATTEINDRE un objet pour le saisir est sans doute la compétence motrice la plus couramment utilisée par les humains. Même sous l'effet de perturbations soudaines et inattendues, les humains semblent réagir rapidement et sont capables d'adapter leurs mouvements en conséquence. Malgré l'apparente facilité et la fréquence avec laquelle nous utilisons cette capacité, il n'est pas possible de prétendre à la compréhension complète des mécanismes sous-jacents). C'est en partie à cause de cela qu'un robot adaptif n'est pas encore parfaitement capable d'atteindre et de saisir les objets en cas de perturbations. Les approches de modélisation des systèmes de contrôle pour la réalisation de ces tâches peuvent être divisées en deux grandes catégories: a) Les approches génératives qui modélisent les mécanismes complexes sous-jacents expliquant la sortie obtenue (dans ce cas le mouvement d'atteindre un objet pour le saisir), et b) Les approches discriminatives qui modélisent directement la sortie observée et imitent le mouvement avec une certaine généralisation. Dans cette thèse, nous adoptons l'approche discriminative pour la modélisation des trajectoires d'atteinte et de saisie d'objets en se basant sur la démonstration du mouvement par un expert. Tout au long de cette thèse, nous emploierons des représentations basées sur des flux indépendants du temps (autonomes) pour apprendre des mécanismes de contrôle de mouvements réactifs qui peuvent ensuite être portés sur les robots.

Cette thèse est divisée en trois parties principales. La première partie est consacrée à la modélisation, biologiquement inspirée, du couplage main-bras durant le mouvement nécessaire à la préhension. Nous nous basons sur des travaux antérieurs dans la modélisation du mouvement à l'aide de systèmes autonomes dynamiques (DS : Dynamical System) et présentons un système dynamique couplé (CDS : Coupled Dynamical System) de ces deux sous-systèmes. Le modèle couplé assure la satisfaction des contraintes entre la main et les sous-systèmes du bras qui sont critiques à la réussite d'une tâche de préhension. En outre, il permet de réduire la complexité du problème global de planification du mouvement par rapport à celui combinant le mouvement de la main et celui du bras en un seul problème.

Dans la deuxième partie, nous étendons l'approche CDS pour prendre en compte plusieurs points de préhension. Un tel modèle est utile vu qu'un bon nombre des objets de la vie quotidienne sont munis de multiples endroits de

préhension sur leur surface. Le choix d’un point de saisie peut être influencé par la position/orientation de l’objet autour de l’extrémité de l’effecteur du robot et ce choix devient critique sous l’effet des perturbations. Si l’objet cible est soudainement perturbé durant l’exécution du mouvement vers une prise (une configuration de préhension) donnée, la nouvelle pose de l’objet peut rendre le point de préhension d’origine inaccessible. Dans un tel scénario, il est souhaitable de passer à un autre point de saisie et d’adopter une nouvelle trajectoire qui lui correspond. Nous combinons une approche basée sur un système dynamique (DS) avec l’apprentissage d’une fonction d’énergie afin d’apprendre un système dynamique à attracteurs multiples dans lequel les attracteurs correspondent à des points de préhension souhaités. Nous présentons le modèle SVM Augmenté (ASVM) qui fusionne la formulation du modèle machine à vecteurs de support (SVM : Support Vector Machine) classique et celle du gradient de contraintes résultantes de la fonction d’énergie afin d’apprendre la fonction dynamique souhaitée de génération de mouvement.

Nous explorons en outre la formulation de l’ASVM comme une technique générale pour l’approximation de fonctions. Nous montrons qu’elle pourrait être utilisée pour l’approximation d’une fonction arbitraire où une combinaison de contraintes d’égalité ou d’inégalité sont spécifiées sur sa valeur ou son gradient pour un ensemble donné de variables d’entrées. D’une manière similaire au SVM classique, nous formulons une version de l’ASVM paramétrée avec le ν qui garantit explicitement une borne inférieure sur le nombre de vecteurs de support obtenus.

Dans la dernière partie de cette thèse, nous abordons le problème de cinématique inverse et d’évitement d’obstacles en combinant notre générateur de mouvement avec des planificateurs globaux dans l’espace de configuration. Nous croyons qu’une technique complète l’autre. D’une part, la nature réactive rapide de notre générateur de mouvement peut être utilisé pour guider la recherche d’un planificateur global du type “randomly exploring random tree (RRT)”. D’une autre part, les planificateurs globaux peuvent gérer efficacement les obstacles arbitraires et éviter les minima locaux présents dans la fonction dynamique apprise par démonstration. Nous montrons que la combinaison de l’information acquise des démonstrations avec la planification globale sous la forme d’une fonction d’énergie diminue considérablement la complexité de calcul de l’état de l’art des planificateurs basés sur l’échantillonnage.

Nous avons évalué nos modèles en simulation et sur des plates-formes robotisées réelles ; le robot humanoïde iCub à 53 degrés de liberté (d.d.l), le bras robotisé KUKA LWR à 7 d.d.l équipé de la main Barrett à 4 d.d.l et la main Allegro à 16 d.d.l. Nous présentons par la suite les contributions de cette thèse aux domaines de la robotique et de l’intelligence artificielle. Tout d’abord, nous avons développé des algorithmes pour la génération de mouvement rapide et adaptatif pour la tâche d’atteinte et de prise d’objets. En second lieu, nous avons formulé une extension à la formulation SVM classique qui prend en compte

le gradient. Nous avons montré que, au lieu d'être limité en tant que classificateur ou un régresseur, SVM peut être utilisé comme une fonction générale pour l'approximation de fonctions. Enfin, nous avons combiné nos méthodes locales avec des approches globales en matière de planification pour éviter les obstacles arbitraires et réduire considérablement la complexité de calcul des planificateurs globaux.

Mots Clé: Mouvements d'atteinte, Le couplage bras-main, Système dynamique non linéaire, Système dynamique couplé, Système dynamique à attracteurs multiples, Apprentissage par imitation, Machine à vecteurs de support, Contraintes de gradient.

*To my beloved parents for their endless love
and the sacrifices they have made.*

*To my wonderful little sister and wife
for their love and support.*

ACKNOWLEDGMENTS

I would like to sincerely thank my supervisor, Aude Billard who extended me the opportunity to conduct my PhD research under her guidance. I deeply appreciate her patience during the early days of my research and carving my rough ideas into something meaningful and scientifically relevant. This thesis would not have been possible without her continuous encouragement, support and critique.

I also acknowledge the examiners, Prof. Alireza Karimi, Prof. Hannes Bleuler, Prof. Jochen Steil and Prof. Ron Alterovitz for their insightful comments and suggestions on an earlier version of the manuscript.

I would like to thank our lab secretaries Karin Elsea and Joanna Erfani and the EDPR secretaries Chabanel Claire and Corinne Lebet for their support with the administrative issues. Many thanks to the EU projects First-MM and AlterEgo for supporting my research.

I am truly thankful to my peers Eric Sauser, Basilio Noris, Elena Gribovskaya, Mohammad Kansari, Florent D'halluin, Mustafa Suphi Erden, Seungsu Kim, Luka Lukic and Guillaume de Chambrier for the scientific discussions which had an important part to play in shaping my work to its current form. Special thanks to Seungsu Kim for his tireless collaboration on the *object catching* research and making us famous!

I feel immensely fortunate to have had such great colleagues at LASA - Sylvain Calinon, Eric Sauser, Basilio Noris, Elena Gribovskaya, Jean-Baptiste Keller, Brenna Argall, Dan Grollman, Florent D'halluin, Martin Duvanel, Mohammad Khansari, Sahar El-Khoury, Seungsu Kim, Luka Lukic, Klas Kronander, Lucia Ureche, Miao Li, Nicolas Sommer, Guillaume de Chambrier, Ajay Tanwani, Ravin De Souza, Silvia Magarelli, Gaetan Cretton, Guillaume Phien, Otpal Vittoz, Hang Yin, Bidan Huang, Nadia Figueroa, Joao Silverio, Joel Ray, Prof. Kenji Tahara, Sina Mirrazavi, Mahdi Khoramshahi, Brahayam Ponton and Iason Batzianoulis. I thank them all for their wonderful friendship and interesting discussions on topics ranging from science, technology, humour to the Indian culture and deep life-lessons.

I would like to thank Guillaume de Chambrier for many things, the least of which is making sure that I had a timely and cheerful lunch. I thank him for being my best buddy in so many ways - close friend, empathiser and a

colleague. Many thanks to - the always smiling and full of positive energy - Sahar El-Khoury for all the joyful discussions and concerns we shared. Thanks to Miao Li and Lucia Ureche for being great companions for our after-five-o-clock fitness activities. Thanks to Suphi Erden for the wonderful time we shared in our office and at our almost ritualistic breaks. Thanks to Klas Kronander for intriguing discussions regarding the Indian cuisine and the occasional guitar lesson. Thanks to Seungsu Kim for all the good times we had on and off work, especially for introducing me to Starcraft! I would finally like to thank my Indian friends for the great times we had outside of work - Abhishek Tewari, Nishanth Ulhas Nair, Brijesh Mishra and his wife Lidija Stankovikj, Aparna Singh, Raja Sundar and Abhishek Kumar.

Lastly, and most importantly, I would like to thank my parents, sister and my wife Anupama for their never ending love and support.

TABLE OF CONTENTS

1	Introduction	1
1.1	Motivation	1
1.2	Literature Review	2
1.2.1	Motion Planning	2
1.2.2	Imitation learning	4
1.3	Contributions	6
1.4	Outline	7
2	Coupled Dynamical System For Hand-Arm Motion Modelling	11
2.1	Foreword	11
2.2	Introduction	11
2.3	Literature Review	14
2.3.1	Manipulation Planning	14
2.3.2	Biological Evidences	15
2.3.3	Imitation Learning	16
2.4	Methodology	17
2.4.1	DS control of reaching	17
2.4.2	Why Coupling Reach and Grasp?	18
2.4.3	Coupled Dynamical System	19
2.5	Experiments and Results	26
2.5.1	Instantiation of CDS variables	27
2.5.2	Validation against human data	27
2.5.3	Validation of the model for robot control	32
2.6	Conclusions	38
3	Multiple Attractor Dynamics For Reach-to-Grasp Motions	41
3.1	Foreword	41
3.2	Introduction	42
3.3	Identifying dynamic constraints	42
3.4	Problem Formulation	45
3.4.1	Primal & Dual forms	46
3.5	Application examples	50
3.5.1	2D Example	50
3.5.2	Error Analysis	51
3.5.3	Sensitivity analysis	52
3.5.4	3D Example	53
3.6	Conclusions	56

4	Augmented-SVM For Gradient Observations In Function Approximation	57
4.1	Foreword	57
4.2	Introduction	57
4.3	Gradient formulations in SVM	59
4.3.1	Partial formulation	60
4.3.2	Complete formulation	61
4.3.3	SMO iterates	63
4.3.4	ν parameterization	66
4.4	Applications	68
4.4.1	Synthetic example	68
4.4.2	Implicit surface reconstruction	70
4.4.3	Energy function learning	71
4.5	Conclusions	72
5	Motion Planning on Energy-Maps Learned From Human Demonstrations	73
5.1	Foreword	73
5.2	Introduction	73
5.3	Literature Review	76
5.4	Energy function learning	77
5.5	Energy-RRT	80
5.6	Hybrid Planning with <i>eRRT</i>	83
5.7	Results	85
5.7.1	Evaluating the basic <i>eRRT</i>	86
5.7.2	Evaluating the hybrid <i>eRRT</i>	87
5.7.3	Planning with goal regions	91
5.8	Conclusions	92
6	Conclusions	95
6.1	Key Contributions	95
6.1.1	Robotics	95
6.1.2	Machine Learning	96
6.2	Limitations and Future Work	97
6.2.1	Generalized coupling in CDS models	97
6.2.2	Global stability in multiple-attractor DS	98
6.2.3	Incremental global planning	98
6.2.4	Robot dynamics consideration	98
	Appendices	101
	Appendix A Appendix	103
A.1	Stability of CDS model	103
A.1.1	CDS model performance with inferred parameters	103
A.2	Kernel Derivatives	104
A.3	Specific kernel expansions	105
A.3.1	RBF Kernel	105
A.3.2	Polynomial Kernel	106
	References	107

INTRODUCTION

1.1 Motivation

Every meaningful action executed by a machine/human agent is aimed at changing the state of the world in a specific way. *Planning* such actions itself requires knowledge of the current state of the world. Real world environments are, however, highly dynamic and unpredictable, making it increasingly difficult to *execute* the planned action and get the desired result. For a robotic agent, the gap between planning and execution proves to be a key hurdle to overcome before it can serve in the real world.

To instantiate these concepts, imagine a simple scenario of reaching over to grasp an object. One may employ a variety of techniques to compute a plan to move the hand towards the object and close the fingers around it. However, a perturbation might occur in several ways: the object moved from its previous location or your arm bumped into something that changed its original trajectory. In the event of such a perturbation, the original plan becomes invalid. The planning-execution paradigm suggests that to recover from such perturbations, the original execution needs to terminate, followed by subsequent re-planning and execution steps. Although such approaches have been a mainstay in robotics, it seems to be far from how effortlessly humans tackle this problem.

Humans exhibit remarkable ability in executing reaching motions and adapting under perturbations. This suggests learning from human demonstrations as a promising avenue for robot motion planning/control. The approaches to tackle this learning problem can be divided into two main categories - *Generative* and *Discriminative*. Generative approaches attempt to model the underlying (usually hidden) process resulting in a specific motor behaviour. Several human physiological studies of reaching motions have pointed out various optimality principles (minimum jerk, energy, noise) that seem to govern the motion generation process. Since these underlying mechanisms are not directly observable, their validity has been judged by their ability to *explain* experimental observations. Although these principles are compact generative models, their optimality, or even feasibility when applied to a specific robotic hardware is questionable. This is due to the kinematic and dynamic differences between a human and robot.

On the other hand, a more direct approach is discriminative modelling where we directly model the desired output independent of the process that might have generated the output. In this thesis, we will develop such models from human demonstrations of reaching motions. We will focus on specific representations of the model that allow fast and online re-planning under unseen perturbations. In the task that we consider in this thesis - reaching with the intention to grasp - perturbations may include displacement of the end-effector and/or the target object (Spatial Perturbations), delays in the task execution due to random factors such as friction or delays in the underlying controller of the robot (Temporal Perturbations) or a change in the target object forcing a change in the grasp location and the type of grasp needed.

For handling spatial perturbations in a reach-to-grasp task, we focus on the following aspects:

1. Adapting the finger pre-shape taking into account the perturbation.
2. Choosing a possibly different grasping point on the object.
3. Avoid arbitrarily shaped obstacles in the environment.

While we tackle the first two problems using purely reactive and online approaches, for the last objective - arbitrary obstacle avoidance - we develop algorithms to leverage the best of both local (reactive) and sampling based global (offline) approaches.

For handling temporal perturbations, throughout this thesis, we will employ a time-invariant flow based model representation. The advantage of such a representation is that it simply maps the current state to the action while ignoring the absolute clock time. This approach, in addition to bridging the planning-execution gap, is also in qualitative agreement with the fact reaching tasks are inherently time-independent. Hence, the classical way of representing motion as timed trajectories should be avoided.

1.2 Literature Review

There is a vast variety of works in point-to-point motion planning that applies to reaching motions. Here we present a broad overview of the current literature in motion planning techniques which in turn motivates the imitation learning paradigm.

1.2.1 MOTION PLANNING

The problem of generating collision-free paths for redundant robotic manipulators has attracted considerable attention in the last two decades (Kavraki et al., 1996a; Canny, 1988; Latombe, 1991; Kuffner and LaValle, 2000; Toussaint, 2009). Broadly, the strategies proposed and pursued in this period can be classified as local and global (Kavraki et al., 1995; Latombe, 1991). Local

methods start from a given initial configuration and step towards the goal configuration using localized information of the workspace. On the other hand, global path planning methods typically apply search algorithms to build a model of the free regions of full state space, which in turn is used to compute a collision free path between two points.

LOCAL METHODS

Local approaches to planning are “greedy” in the nature of their search for a feasible path. Typical formulations have been around articulated *potential-field* functions guiding the search along the flow of the (negative) gradient (Khatib, 1986; Khosla and Volpe, 1988; Koditschek, 1989; Barraquand and Latombe, 1991). These methods are quite effective for geometrically simple state spaces in their ability to react online toward perturbations. This ability results by construction of these methods, i.e., that they do not generate full paths but recommend an action - end-effector velocity, joint torques etc. - to be taken in the current state. Although suitable for online reactive motion generation, they are prone to local minima in the potential field. Certain local-minimum free formulations employing harmonic potential fields have been presented by Koditschek (1987); Rimón and Koditschek (1992); Barraquand et al. (1992). However, their utility remains confined to low dimensional state spaces due to large computational effort required for computing the potential functions. To reduce the computation complexity, approximate potential functions have been proposed which are essentially piecewise-continuous functions defined over a grid in the state space. Although these can be considered as a hybrid between local and global planning methods, they still rely on re-computation of the potential function in case of perturbations, and hence not suitable for reactive planning.

In the first part of this thesis, we present algorithms that retain the desirable properties of local methods - reactivity and online adaptation - while not relying on exhaustive computation or re-computation under perturbations.

GLOBAL METHODS

A prominent class of work in global planning methods deals with randomized sampling of the state space. Randomly exploring random trees (RRT) presented by Kuffner and LaValle (2000) build a map of the collision free state space by efficient sampling using Voronoi bias. An attractive property of such approaches is *probabilistic completeness* which implies that the probability of finding a feasible path decreases monotonically as more time is given to the planner. Although this is extremely useful in cases where the free space is non-convex, they are typically designed to work in a plan-execute manner which makes them difficult to use in dynamically changing environments. A similar line of work - *probabilistic roadmaps* (PRM) - by Kavraki et al. (1996a) comprised of a learning phase and a query phase. A connectivity map is built during the learning phase

by randomly sampling the space and connecting the collision free nodes using a local planner. Due to the separate learning phase, queries are processed much faster. However the time required by the learning phase to represent the state space sufficiently well can be large. Garber and Lin (2004); Gayle et al. (2007) presented heuristic methods which proved to be better than completely randomized techniques. A recent body of works (Tedrake et al., 2010; Brock et al., 2008) falls in the category of *feedback planning* which provide certain level of re-planning ability thereby partly alleviating the problem with global methods. However, they share the drawback of being computationally expensive.

A common property among all global approaches is the ability to handle arbitrary shaped obstacles and workspaces, albeit at a high computational cost. Chapter 5 of this thesis presents a combination of a sampling based global planning approach with our online motion learning framework. Results show that such a combination enables the algorithm to tackle arbitrary workspaces while substantially alleviating the major drawback with global methods i.e., computational complexity.

1.2.2 IMITATION LEARNING

A separate body of works re-visits the problem of motion planning as one of learning from data. These methods take inspiration from the way humans learn new skills from other experts and aim at transferring skills (in the form of motion plans) from experts to robots. All the information about the task and the global workspace is embedded within a set of demonstrations which may be provided, for instance, by back-driving the robot’s joints or through teleoperation. In these approaches, the underlying policy that generates the motion is only implicitly specified in the form of demonstrations. This implicit representation is often easier and more intuitive to specify which makes learning based methods an attractive choice for motion planning.

A set of key questions have been identified in early studies which should be addressed by imitation learning approaches, namely, *what / how / when / who* to imitate? (Nehaniv and Dautenhahn, 1999). In this thesis we focus on the question of *what* to imitate which refers to the problem of choosing a representation for specifying the imitation learning problem as a machine learning problem and give a solution to it. Classical approaches addressing this question have focussed on a) symbolic encoding for high level tasks (Muench et al., 1994; Saunders et al., 2006) and b) trajectory learning for low level motion generation (see Argall et al. (2009); Schaal et al. (2003); Calinon (2008) and references therein). Although there is a large volume of literature on each, here we present an overview of the low level approaches which is also the focus of this thesis.

A variety of methods have been used for encoding the demonstrations at a trajectory level. Spline based methods (Miyamoto et al., 1996; Andersson, 1989)

provide a fast method for learning motions from a set of demonstrated trajectories. Regression based approaches have been presented in several works that reduce the learning problem to that of constrained minimization (Yamane et al., 2004; Atkeson et al., 1997). In order to better handle noise and uncertainty in real-world data, statistical learning techniques have been developed that provide a richer class of algorithms to model complex behaviors. Prior work from our lab by Calinon et al. (2007); Calinon and Billard (2007) used Gaussian Mixture Models (GMM) to learn a probabilistic representation of the constraints inherent in a task. A variety of other machine learning models have been used to encode trajectories, e.g., Hidden Markov Models (HMM) (Inamura et al., 2003; Calinon and Billard, 2005), neural networks Reinhart and Steil (2011), fuzzy logic (Berenji and Khedkar, 1992) etc. This thesis follows up on this line of work by employing machine learning techniques - coupled GMM and a novel SVM based learning framework - for modelling a set of noisy demonstrations of reach-to-grasp motions.

Many recent approaches have moved away from modelling trajectories and focussed on learning dynamical systems (DS) underlying the demonstrations. In this thesis also, we advocate the use of DS as a low-level representation of the task due to its various desirable properties. The next subsection presents motivating arguments for our choice and reviews other DS based learning approaches used in robot motion planning.

DYNAMICAL SYSTEMS FOR IMITATION LEARNING

Many physiological studies (Hoffmann, 2011; Bullock and Grossberg, 1988b) have implicated DS for motion generation in humans. These results have inspired their use as an alternative to classical motion planning algorithms in robotics. Use of such models alleviates the planning-execution gap and views them as coupled problems wherein motion is generated by a DS evolving in time. A major advantage of representing motion using DS based models is the ability to counter perturbations by virtue of the fact that re-planning of trajectories is instantaneous. Secondly, autonomous DS can be employed to avoid explicit representation of time in the modelling which provides a more human-like adaptation in case of perturbations.

A variety of data-driven methods have been employed to realize motion planning in the form of dynamical systems. Many recent works (Ijspeert et al., 2001; Ude et al., 2010; Neumann et al., 2013; Wolpert and Kawato, 1998; Gribovskaya et al., 2011) have used DS to encode from a set of trajectories. Model-free approaches provided by k-nearest neighbors Moore (1990), Gaussian processes Wang et al. (2005) and locally weighted regression (LWR) Atkeson et al. (1997) have been quite popular due to the absence of assumptions about the data. Reservoir based techniques developed by Neumann et al. (2013); LukosEvičius and Jaeger (2009) cast the problem of learning as a linear regression while still achieving non-linear output. Although model free methods are the most general

approximators, gathering enough data to achieve good performance is difficult in high dimensions. On the other hand, model based methods have proved to be useful for learning with only a few demonstrations. [Dixon and Khosla \(2004\)](#) use a linear approximation and fit model parameters using the observed data. Although this ensures a stable output motion, the linearity assumption is too restrictive for many tasks. [Hersch et al. \(2008\)](#); [Ijspeert et al. \(2001\)](#) learn a non-linear modulation of a stable DS typically modeled as a spring-damper system. Time series based methods ([Tong, 1990](#); [Chamroukhi et al., 2013](#)) have been used to build a local model of the dynamics. Numerical approaches based on splines ([Lee, 1986](#)) and radial basis functions ([Elanayar et al., 1994](#)) have been extensively used. However, parameter tuning remains a persistent problem with these methods. [Gribovskaya et al. \(2011\)](#); [Khansari Zadeh and Billard \(2010a\)](#) formulated the problem as a non-linear mixture of several linear systems using a GMM and learned an autonomous dynamical system (DS) that best explains the demonstrations. The learned DS was then used for online and reactive motion generation. This technique was further refined in [Khansari Zadeh and Billard \(2011b\)](#); [Mohammad Khansari Zadeh and Billard \(2014\)](#) to ensure stability of the learned DS.

This thesis builds on state-of-the-art methods in DS based motion learning by presenting enhancements, i.e., coupled DS and multiple-attractor DS, designed for generating reach-to-grasp motions in arbitrary environments. These contributions are detailed in the next section.

1.3 Contributions

This thesis addresses the problem of encoding and generating motions for robots under real-time perturbations. We specifically consider a particular class of motions - reaching with the intent to grasp. We address the following aspects of this problem:

Hand-arm coupling One of the well known phenomena in human reach-to-grasp motions is the inherent coupling between the motion of hand and arm control systems. This part of the thesis focuses on developing quantitative models backed by experimental data of human reach-to-grasp motions under perturbations. We empirically show that there exists a difference in the dynamics employed by human subjects during perturbed and unperturbed demonstrations. We show that successful task completion is ensured by this coupling which exists between the reach and grasp components. We further present a coupled dynamical systems based approach to achieve the observed coordination between hand transport and pre-shape. The DS based formulation enables our model to react under very fast on-the-fly perturbations without any latency for re-planning.

Multiple goals Many real-world scenarios of reaching and grasping involve

several grasping points. We extend the previously presented DS based motion modelling approach to incorporate these multiple grasping points as different attractors of a dynamical system. In effect, we propose a modifier to the original dynamical systems which learns the switching strategy between the different attractors and while maintaining local stability at each of the attractors. The modifier function is learned from a novel reformulation of the classical support vector machine framework. We further develop this framework as a general function approximation tool which allows information about the function gradient - in the form of equality or inequality constraints - to be incorporated within the same optimization formulation. We show that the learning problem can be presented (in the dual form) as a convex quadratic program whose globally optimal solution can be obtained using fast sequential minimal optimization (SMO) (Platt, 1998) like updates.

Non-convex obstacles Many existing approaches combine reactive motion planning with obstacle avoidance where the explicit shape of the obstacle in the planner space is known and convex. However for redundant manipulators in real environments, the assumption of convexity is too restrictive. We propose a combination of state-of-the-art global sampling based planners with flow based models learned from demonstrations to leverage the best of both techniques. Such a combination implies the following advantages: a) Arbitrary robot geometries and obstacles can be handled without any special treatment or preprocessing and b) Information from demonstrations can be used to guide sampling based planners and hence significantly decrease the computational burden.

1.4 Outline

This thesis has been divided into Chapters as follows.

Chapter 2

Coupled Dynamical System For Hand-Arm Motion Modelling

This chapter starts with a brief introduction to Dynamical Systems (DS) and their estimation using Gaussian Mixture Models which is then extended to the Coupled Dynamical System (CDS) model for motion modelling and generation. We perform experiments to learn from perturbed human demonstrations and validate our approach by presenting a series of experiments on the iCub simulator as well as the real robot. We show that our model reproduces the motions while respecting the correlations and couplings learned from the demonstrations which are critical for the success of the overall task. We also show that the post-perturbation re-planning is quick and enables very fast response from the robot.

Chapter 3

Multi-Attractor Dynamical System For Multiple Goal Reaching Motions

This chapter extends the previously presented flow based motion planning algorithm to handle multiple grasping locations modelled as different attractors of a dynamical system. To this end, we learn a modulation function which ensures that the desired properties of a multiple-attractor DS are retained in the model. These properties are the following: a) Ensuring strict classification across the regions of attraction (ROA) of each DS, b) Following closely the dynamics of each DS in each of the ROA and c) Ensuring that all trajectories in each ROA terminate at the desired attractor. Satisfying requirements a) and b) is equivalent to performing classification and regression simultaneously. We take advantage of the fact that the optimization in support vector classification and support vector regression have the same form to phrase our problem in a single constrained optimization framework called *Augmented-SVM* (ASVM). The model is validated on a set of reaching experiments with the KUKA LWR arm in simulation and on the real platform. We show that the robot is able to switch online and seamlessly between the different attractors with extremely low latency. This is demonstrated by an experiment where the robot catches a falling object while selecting the correct grasping point in real-time.

Chapter 4

Augmented-SVM For Gradient Observations In Function Approximation

In this chapter we generalize the previously presented ASVM formulation. We present a general framework for function approximation which makes the formulation in the previous chapter a particular case. The framework developed in this chapter can combine function gradient information with the function value at several input locations. The information may be provided in the form of equality constraints - as is done in regression problems - or in the form of inequality constraints - as is done in classification problems. While existing Support Vector Machine formulations aim at learning either a classifier (inequality constraints) or a regressor (equality constraints), our framework combines both inequality and equality constraints simultaneously within the same optimization problem.

Chapter 5

Motion Planning on Energy-Maps Learned From Human Demonstrations

In this chapter we propose a combination of local methods presented before and state-of-the-art global planners to achieve obstacle avoidance in arbitrary environments. The combination strategy involves learning an energy function from a set of demonstrations of reaching motions and use this to inform the search in a sampling based planning algorithm. The special properties of the energy function expedite the sampling based search by restricting exploration

only to regions where it is needed and suggesting directions for search based on the demonstrations. We present a randomly exploring random tree based algorithm to search for low-energy paths over the learned map. We compare our method with existing sampling based planners that plan over cost-maps. Results show that planning over energy-maps rather than cost-maps results in faster planning times and comparable path-costs.

Chapter 6

Conclusions and Discussion

In this chapter we present a summary of this thesis, outline the key contributions and discuss avenues for future work.

COUPLED DYNAMICAL SYSTEM FOR HAND-ARM MOTION MODELLING

2.1 Foreword

Under the general theme of this thesis, i.e., modelling reach-to-grasp motions, in this chapter we focus on the problem of controlling the hand and finger motions in a synchronized manner. Several physiological experiments have suggested an inherent hand-arm coupling in humans. In this chapter we develop a model and corresponding learning scheme to capture this coupling from a set of demonstrations. The proposed model here builds upon state-of-the art in dynamical systems based motion planning and extends it by introducing a coupling between two different dynamical systems. This work lead to the following publications (in reverse chronological order):

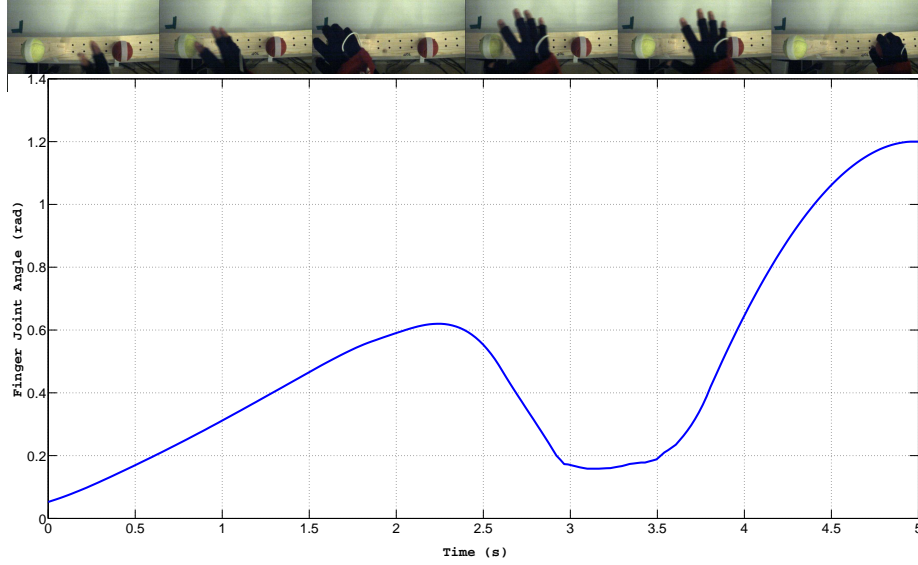
- A. Shukla and A. Billard. Coupled dynamical system based arm-hand grasping model for learning fast adaptation strategies under real-time perturbations. In *Proceedings of Robotics: Science and Systems VII*, pages 313 – 320. MIT Press, 2011. ISBN 978-0-262-51779-9
- A. Shukla and A. Billard. Coupled dynamical system based arm-hand grasping model for learning fast adaptation strategies. *Robotics and Autonomous Systems*, 60(3):424 – 440, 2012a. ISSN 0921-8890. doi: 10.1016/j.robot.2011.07.023
- S. Kim, A. Shukla, and A. Billard. Catching objects in flight. *Robotics. IEEE Transactions on*, 2014

2.2 Introduction

From a planning/control perspective, modelling constrained grasping motions have often been studied as two separate problems in which one first generates the arm motion (Berenson et al., 2009) and then shapes the hand to grasp stably the targeted object (Miller et al., 2003; Bone et al., 2008). The sheer complexity of each of these two problems when controlling high dimensional arm-hand systems has discouraged the use of a single coherent framework for carrying out both tasks simultaneously. In this work, we advocate the use of a single framework to control reach and grasp motion when the task requires very



(a) Experimental Setup



(b) Hand Closeup

Figure 2.1: (a) - Experimental setup to record human behavior under perturbations. On screen target selector is used to create a sudden change in the target location for reaching. (b) - Motion of the fingers as seen from a high speed camera @ 100 fps. Note the decrease in the joint angle values (re-opening of fingers) starting at the onset of perturbation.

fast adaptation of the motion. We consider the problem of on-the-fly replanning reach and grasp motion for enabling adaptation to changes in the position, size or type of object to be grasped. This requires the ability for fast and flexible re-planning.

One widely desired property when designing a robot controller is robustness, i.e. the ability to robustly recover from perturbations. In reach-to-grasp tasks, perturbations may be of the following types: a) displacement of the robot end-effector and/or the target (spatial perturbations), b) delays in the task execution due to random factors such as friction in the gears or delays in the underlying controller of the robot (temporal Perturbations), c) change in the target object forcing a change in the type of the grasp required. In the context of controlling for reach and grasp tasks, this problem has been addressed primarily by designing a stable controller (ensured to stop at the target) for both reach and grasp components of motion. Many different ways have been offered to designing task-specific controllers with minimum uncertainties and deviations from the intended trajectory. One drawback of such approaches is that they assume that the trajectory to track is known before hand. It is how-

ever not always desirable to return to the original desired trajectory as the path to get there may be infeasible, especially when a perturbation sent us far from the originally planned trajectory. More recent approaches have advocated the use of dynamical systems as a natural means of embedding sets of feasible trajectories. This offers great robustness in the face of perturbations, as a new desired trajectory can be recomputed on the fly with no need to re-plan (Pastor et al., 2008; Dan and Todorov, 2009; Grimes et al., 2007). We follow this trend and extend previous works done in our lab by Gribovskaya and Billard (2009); Khansari Zadeh and Billard (2010b) on learning a motor control law using time-invariant dynamical systems. Such a control law generates trajectories that are asymptotically stable at a single attractor. In this chapter, we extend this work to enable coupling across two such dynamical systems for controlling reach and grasp motions in synchrony. Controlling for such coupled dynamical systems entails more complexity than controlling using two independent control laws to ensure satisfaction of convergence constraints and correlations between the two processes (Castiello et al., 1998; Gentilucci et al., 1992; Jakobson and Goodale, 1991; Jeannerod, 1984).

We follow a Programming by Demonstration (PbD) approach (Billard et al., 2008) and investigate how we can take inspiration from the way humans react when perturbed and learn motor control laws from such examples. This departs from the usual approaches in PbD that usually use demonstrations of *unperturbed* motions.

A number of studies of the way humans, and other animals, control reach-and-grasp tasks (Paulignan et al., 1991; Castiello et al., 1993, 1998) have established that the dynamics of arm and finger movements follow a particular pattern of coordination, whereby the fingers start opening (*preshape*) for the final posture at about half of the reaching cycle motion. Humans and other animals adapt both the timing of hand transport and the size of the finger aperture to the object’s size and location. When perturbed, humans adapt these two variables seamlessly and in synchrony (Kawato, 1999; Engstrom and Kelso, 2008). Christel and Billard (2001) showed a strong coupling between the dynamics of finger aperture and the hand velocity. Finger aperture is composed of a biphasic course, i.e. a short and wide opening after hand peak velocity is followed by a slow closure phase. In this chapter, we revisit these observations to derive precise measurement of the correlation between hand transport and fingers preshape, which we then use to determine specific parameters of our model of coupled dynamical systems across these two motor programs.

This chapter is structured as follows. Section 2.3 reviews the literature related with the presented work: imitation learning, manipulation planning and biological evidences of reach-grasp coupling. Section 2.4 starts with a short recap of the background of Dynamical Systems (DS), their estimation using GMMs and performing regression. We give a formal definition of the Coupled Dynamical System (CDS) model, explain the model construction process and

give the algorithm for regression. In [Section 2.5](#), we present the experimental setup used to learn from perturbed human demonstrations. We validate our approach by presenting a series of experiments on the iCub simulator as well as the real robot. We show that the reach-grasp behaviour is reproduced while respecting the correlations and couplings learned during the demonstrations and that it is critical for the success of the overall task. We also show that the post-perturbation re-planning is quick and enables very fast response from the robot.

2.3 Literature Review

The presented model relates to different fields of work. It draws inspiration from neuro-physiological studies of human reach-to-grasp motion, exploits the current techniques from imitation learning to add novel contribution in the field of manipulation planning and control. In this section, we review the relevant literature in each of these fields.

2.3.1 MANIPULATION PLANNING

The classical approach in robotics for reaching to grasp objects has been to divide the overall problem into two sub-problems, where one first reaches for and then grasps the objects ([Berenson et al., 2009](#); [Vahrenkamp et al., 2009](#); [Harada et al., 2008](#)). Although both the issues of reaching to a pre-grasp pose and formation of grasp around arbitrary objects are intensively studied, very few ([Bae et al., 2006](#); [Gienger et al., 2008](#); [Hsiao et al., 2009](#)) have looked into combining the two so as to have a unified reach-grasp system.

Most manipulation planners typically plan paths in the configuration space of the robot using graph based techniques. Very powerful methods such as those based on probabilistic roadmap and its variants ([Saut et al., 2007](#); [Gasparri et al., 2009](#)) use a C-space description of the environment and graph based methods for search. Another approach to the same problem has been adopted by using various control schemes in conjunction with offline grasp planners ([Harada et al., 2008](#)) or visual tracking systems ([Morales et al., 2007](#)). A synergistic combination of grasp planning, visual tracking and arm trajectory generation is presented in ([Kragic et al., 2001](#)). [LaValle and Kuffner \(2001\)](#) proposed RRT's as a faster alternative to manipulation planning problems, provided the existence of an efficient inverse kinematic (IK) solver. RRT based methods ([Berenson et al., 2009](#)) are currently the fastest online planners due to their efficient searching ability. The reported planning times are of the order of 100 ms for single arm reaching tasks in the absence of any obstacles ([Vahrenkamp et al., 2009](#)). While this is certainly very quick, graph based methods lose to take into account the dynamic constraints of the task.

It remains a challenge to design planning algorithms for dynamic tasks under quick perturbations. Moreover, in such cases, re-planning upon perturbation

must not take more than a few milliseconds. These are the type of problems we address here. We take inspiration from human studies to understand how humans embed correlations between arm and finger motion to ensure robust response to such fast perturbations. We show that retaining these correlations between the reach and grasp motions is critical to the success of the tasks.

2.3.2 BIOLOGICAL EVIDENCES

The concept of coupling between the reach and grasp motions is inspired by extensive evidence in neuro-physiological studies (Jeannerod, 1984; Vilaplana et al., 2005; Gentilucci et al., 1991; Saling et al., 1996; Castiello et al., 1998; Mitz et al., 1991; Meulenbroek et al., 2001; Paulignan et al., 1994). The most frequently reported mechanism suggests a parallel, but time-coupled evolution of the reach and grasp motions with synchronized termination. Attempts at quantifying this process in a way that may be usable for robot control are few. Gulke et al. (2010); Bae and Armstrong (2011) showed that the finger motion during reach to grasp tasks could be described by a simple polynomial function of time, while Ulloa and Bullock (2003) modeled the covariation of the arm and the finger motion. Interestingly, these authors also report on an involuntary reopening of the fingers upon perturbation of the target location; an observation which we will revisit in this chapter.

A number of models have been developed to simulate the finger-hand coupling, so as to account for the known coordinated pattern of hand-arm motions. The Hoff-Arbib model (Hoff and Arbib, 1993) generates a heuristic estimate of the transport time based on the reaching distance and object size and uses it to compute the opening and closing times of the hand. Since the control scheme presented in their approach was time dependent and the temporal coupling parameters decided prior to the onset of movement, this model did not guarantee handling of temporal or spatial perturbations. Oztop and Arbib (2002) argued in their *hand state hypothesis* that during human reach-grasp motion control, the most appropriate feedback is a 7 dimensional vector, including pose of the hand w.r.t the target, hand aperture and thumb adduction/abduction. In the Haggard-Wing model (Haggard and Wing, 1995), both processes of transport and aperture control have access to each other’s spatial state. The variance of hand and finger joint angles is used to set the corresponding control gains, whereas computation of the correlation across the two is used to implement the spatial coupling. The time independency of this model proved to be an elegant way to handle temporal perturbations. A neural network based model presented by Ulloa and Bullock (2002) ensured continuous coupling and efficient handling of perturbations. They assumed the *vector integration to end point* (VITE) model (Bullock and Grossberg, 1988a) as a basis for task dynamics.

The model we propose here specifically exploits the principle of spatial coupling between the palm and finger motion. It ensures that the motion repro-

duced by the robot exhibits hand-arm coupling and respects termination constraints similar to what are found in natural human motion.

2.3.3 IMITATION LEARNING

Learning how to perform a task by observing demonstrations from an experienced agent has been explored extensively under different frameworks. Classical means of encoding the task information are based on spline or polynomial decomposition and averaging (Hwang et al., 2003; Keshmiri et al., 2010). These have been shown to be very fast trajectory generators, useful for tasks like catching moving objects. A different body of work advocates non-linear stochastic regression techniques in order to represent the tasks and regenerate motion in a generalized setting (Schaal and Atkeson, 1998). These methods allow systematic treatment of uncertainty by assuming data noise and hence *estimate* the trajectories as a set of random variables. The regression assumes a model for the underlying process and learns its parameters via machine learning techniques. Subsequently, multiple works under the PbD framework (Schaal et al., 2000; Ijspeert et al., 2002; Gribovskaya and Billard, 2009) have shown that this problem can be handled elegantly by using the dynamical systems (DS) approach. Using DS to represent motion removes the explicit time dependency from the model. As a result, transitions between the states during the execution of a task depend solely on the current state of the robot and the environment¹. However, the removal of time dependency is introduced at the cost of non-trivial stability of the models. The states of a process evolving autonomously under the influence of a DS may diverge away from the goal if initialized outside the basin of attraction of the equilibrium point. Eppner et al. (2009) presented a dynamic Bayesian networks based approach to learn generalized relations between the world and the robot/demonstrator. While generalizing the task reproduction over different spatial setups, this framework also allows to include constraints not captured from the demonstrations, such as obstacle avoidance. Ijspeert et al. (Ijspeert et al., 2002) in their *dynamic movement primitives* (DMP) formulation, augment the dynamics learned from motion data with a stable linear dynamics which would take precedence as the state reaches close to the goal. In previous work (Khansari Zadeh and Billard, 2010b), it was shown that formulating the problem of fitting data to the Gaussian Mixture Model as a non-linear optimization problem under stability constraints ensures global asymptotic stability of the DS.

Although a large amount of work has been done on learning and improving skills from observing good examples of successful behavior, very few work has looked into the information that can be extracted from the non-canonical demonstrations. In Grollman and Billard (2011), we proposed one way to learn from failed demonstrations. Here, we follow a complementary road and inves-

¹Even in a DS formulation, time dependency is present but only implicitly in the form of time derivatives of the state variables.

tigate how one can learn from observing how humans adapt their motion so as to avoid failure. To the best of our knowledge, this is the first work on PbD that studies human motion recovery under perturbation. We empirically show - a) the difference in dynamics employed during perturbed and unperturbed demonstrations and b) the coupling that exists between the reach and grasp components which ensures successful task completion. We present a coupled dynamical systems based approach to achieve coordination between hand transport and pre-shape. We show that the DS based formulation enables our model to react under very fast on-the-fly perturbations without any latency for re-planning. We validate the model by implementing our method on the iCub simulator as well as the real robot.

2.4 Methodology

In this section, we start with a short description of motion encoding using autonomous dynamical systems (DS) and explain how Gaussian Mixture Models (GMMs) can be used to estimate them. We then present an extension of this GMM estimate to allow coupling across different DS, which we further refer to as Coupled Dynamical System (CDS). A formal discussion of the Coupled Dynamical System (CDS) model is presented describing the modeling process and regression algorithm to reproduce the task and a simple 2D example is included to establish intuitive understanding of the working of the CDS model.

2.4.1 DS CONTROL OF REACHING

We here briefly present our previous work on modeling reaching motion through autonomous dynamical systems with a single attractor at the target. For clarity, we reiterate the encoding presented in [Gribovskaya and Billard \(2009\)](#).

Let ξ denote the end-effector position and $\dot{\xi}$ its velocity. We further assume that the state of the system evolves in time according to a first order autonomous Ordinary Differential Equation (ODE):

$$\dot{\xi} = \mathbf{f}(\xi) \quad (2.1)$$

$\mathbf{f} : \mathbb{R}^d \mapsto \mathbb{R}^d$ is a continuous and continuously differentiable function with a single equilibrium point at the attractor, denoted ξ^* and we have:

$$\lim_{t \rightarrow \infty} \xi(t) = \mathbf{0} \quad (2.2)$$

We do not know \mathbf{f} but we are provided with a set of N demonstrations of the task where the state vector and its velocities are recorded at particular time intervals, yielding the data set $\{\xi_n^t, \dot{\xi}_n^t\} \forall t \in [0, T_n]; n \in [1, N]$. T_n denotes

the number of data points in demonstration n . We assume that this data was generated by our function \mathbf{f} subjected to a white Gaussian noise ϵ and hence we have:

$$\dot{\boldsymbol{\xi}} = \mathbf{f}(\boldsymbol{\xi}; \boldsymbol{\theta}) + \epsilon \quad (2.3)$$

Notice that \mathbf{f} is now parameterized by the vector $\boldsymbol{\theta}$, that represents the parameters of the model we will use to estimate \mathbf{f} .

We build an estimate of the function $\hat{\mathbf{f}}$ in two steps. We first build a probability density model of the data by modeling it through a mixture of K Gaussian functions. The core assumption when representing a task as a Gaussian Mixture Model (GMM) is that each recorded point $\boldsymbol{\xi}(t)$ from the demonstrations is a sample drawn from the joint distribution:

$$\boldsymbol{\xi} \sim \mathcal{P}(\boldsymbol{\xi}, \dot{\boldsymbol{\xi}} | \boldsymbol{\theta}) = \sum_{k=1}^K \pi^k \mathcal{N}(\boldsymbol{\xi}, \dot{\boldsymbol{\xi}}; \boldsymbol{\theta}^k) \quad (2.4)$$

with $\mathcal{N}(\boldsymbol{\xi}; \boldsymbol{\theta}^k) = \frac{1}{\sqrt{(2\pi)^{2d} |\Sigma^k|}} e^{\frac{1}{2}(\boldsymbol{\xi} - \boldsymbol{\mu}^k)^T (\Sigma^k)^{-1} (\boldsymbol{\xi} - \boldsymbol{\mu}^k)}$ and where π^k , $\boldsymbol{\mu}^k$ and Σ^k , are the component weights, means and covariances of the k -th Gaussian.

Taking then the posterior mean estimate of $\mathcal{P}(\dot{\boldsymbol{\xi}} | \boldsymbol{\xi})$ yields a noise-free estimate of our underlying function:

$$\dot{\boldsymbol{\xi}} = \sum_{k=1}^K h^k(\boldsymbol{\xi}) (\mathbf{A}^k + \mathbf{b}^k) \quad (2.5)$$

where,

$$\left. \begin{aligned} \mathbf{A}^k &= \Sigma_{\dot{\boldsymbol{\xi}} \boldsymbol{\xi}}^k (\Sigma_{\boldsymbol{\xi}}^k)^{-1} \\ \mathbf{b}^k &= \boldsymbol{\mu}_{\dot{\boldsymbol{\xi}}}^k - \mathbf{A}^k \boldsymbol{\mu}_{\boldsymbol{\xi}}^k \\ h^k(\boldsymbol{\xi}) &= \frac{\pi^k \mathcal{N}(\boldsymbol{\xi}; \boldsymbol{\theta}^k)}{\sum_{i=1}^K \pi^i \mathcal{N}(\boldsymbol{\xi}; \boldsymbol{\theta}^i)} \end{aligned} \right\}. \quad (2.6)$$

To ensure that the resulting function is asymptotically stable at the target, we use the *stable estimator of dynamical system* (SEDS) approach, see [Khansari Zadeh and Billard \(2010b\)](#) for a complete description. In short, SEDS determines the set of parameters $\boldsymbol{\theta}$ that maximizes the likelihood of the demonstrations being generated by the model, under strict constraints of global asymptotic stability. Next, we explain how this basic model is exploited to ensure that the hand and fingers reach the target even when perturbed. We further show how it is extended to build an explicit coupling between hand and finger motion dynamics to ensure robust and coordinated reach and grasp.

2.4.2 WHY COUPLING REACH AND GRASP?

We are now facing the problem of extending our reaching model presented in the previous section to allow successful hand-arm coordination when performing

reach and grasp. The scheme presented in our previous works, as explained in previous sub-section, assumes that the motion is point-to-point in high dimensional space. As a result, the state vector ξ converges uniformly and asymptotically to the target. To perform reach-grasp tasks, such a scheme could be exploited in two ways. One could either:

1. Learn two separate and independent DS with state vectors as end-effector pose and finger configurations.
2. Or learn one DS with an extended state vector consisting of degrees of freedom of the end-effector pose as well as finger configurations.

Learning two DS would not be desirable at all since then two sub-systems (transport and pre-shape) would evolve independently using their respective learned dynamics. Hence, any perturbation in hand transport would leave the two sub-systems temporally out of synchronization. This may lead to failure of the overall reach-grasp task even when both the individual DS will have converged to their respective goal states.

At first glance, the second option is more appealing as one could hope to be able to learn the correlation between hand and finger dynamics, which would then ensure that the temporal constraints between the convergence of transport and hand pre-shape motions will be retained during reproduction. In practice, good modeling of such an *implicit* coupling in high-dimensional system is hard to ensure. The model is as good as the demonstrations are. If one is provided with relatively few demonstrations (in PbD one targets less than ten demonstrations for the training to be bearable to the trainer), chances are that the correlations will be poorly rendered, especially when querying the system far away from the demonstrations. Hence, if the state of the robot is perturbed away from the region of the state space which was demonstrated, one may not ensure that the two systems will be properly synchronized. We will establish this by the means of a simulation experiment in [Section 2.5](#).

We here take an intermediary approach in which two separate DS are first learned and then coupled explicitly. In the context of reach-and-grasp tasks, the two separate DS correspond to the hand transport (dynamics of the end-effector motion) and the hand pre-shape (dynamics of the finger joint motion). We will assume that the transport process evolves independently of the fingers' motions while the instantaneous dynamics followed by the fingers depends on state of the hand. This will result in the desired behavior, namely that the fingers will reopen when the object is moved away from the target. Note that the finger-hand coupling will be parameterized. We will show in the experiments that this coupling can be tuned by changing the model parameters to favor either "human-like" motion or fast adaptive motion to recover from quick perturbations.

2.4.3 COUPLED DYNAMICAL SYSTEM

In the following subsections, we present the formalism behind the Coupled Dynamical System (CDS), describing how we learn the model and how we then query the model during task execution. To facilitate understanding of the task reproduction using the CDS model, we illustrate the latter in a 2D example that offers a simplistic representation of the high-dimensional implementation presented in the result section.

CDS MODEL

Let $\xi_x \in \mathbb{R}^3$ denote the cartesian position of the hand and $\xi_f \in \mathbb{R}^{d_f}$ the joint angles of the fingers. d_f denotes the total number of degrees of freedom of the fingers. The hand and the fingers follow separate autonomous DS with associated attractors. For convenience, we place the attractors at the origin of the frames of reference of both the hand motion and the finger motion and hence we have: $\xi_x^* = 0$ and $\xi_f^* = 0$. In other words, the hand motion is expressed in a coordinate frame attached to the object to be grasped, while the zero of the finger joint angles is placed at the joint configuration adopted by the fingers when the object is in the grasp. We assume that there is a single grasp configuration for a given object. Since the reach and grasp dynamics may vary depending on the object to be grasped, we will build a separate CDS model for each object considered here. We denote the set \mathcal{G} of all objects for which grasping behaviors are demonstrated.

The following three joint distributions, learned as separate GMMs, combine to form the CDS model:

1. $\mathcal{P}(\xi_x, \dot{\xi}_x | \theta_x^g)$: encoding the dynamics of the hand transport
2. $\mathcal{P}(\Psi(\xi_x), \xi_f | \theta_{inf}^g)$: encoding the joint probability distribution of the inferred state of the fingers and the current hand position
3. $\mathcal{P}(\xi_f, \dot{\xi}_f | \theta_f^g)$: encoding the dynamics of the finger motion

$\forall g \in \mathcal{G}$. Here, $\Psi : \mathbb{R}^3 \mapsto \mathbb{R}$ denotes the *coupling function* which is a monotonic function of ξ_x satisfying:

$$\Psi(\mathbf{0}) = 0. \quad (2.7)$$

This function should be specified by taking into account the nature of the task. θ_x^g , θ_f^g and θ_{inf}^g denote the parameter vectors of the GMMs encoding the hand-transport dynamics, finger motion dynamics and the inference model respectively.

The distributions $\mathcal{P}(\xi_x, \dot{\xi}_x | \theta_x^g)$ and $\mathcal{P}(\xi_f, \dot{\xi}_f | \theta_f^g)$ that represent an estimate of the dynamics of the hand and finger motion respectively are learned using the same procedure as described in [Section 2.4.1](#). To recall, each density is modeled through a mixture of Gaussian functions. As explained in [Section 2.4.1](#), in order to ensure that the resulting mixture is asymptotically stable at

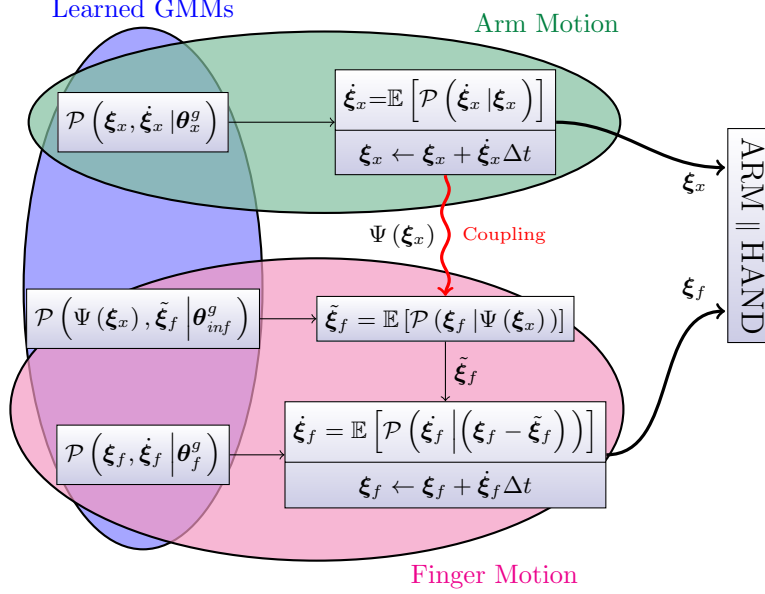


Figure 2.2: Task execution using CDS model. Blue region shows the three Gaussian Mixture Models which form the full CDS model. Green region shows the sub-system which controls the dynamics of the hand transport. Magenta region shows the sub-system controlling finger motion, while being influenced by the state of the hand transport sub-system. Coupling is ensured by passing selective state information in the form of $\Psi(\xi_x)$ as shown in red.

the attractor (here the origin of each system), we use the SEDS learning algorithm [Khansari Zadeh and Billard \(2010b\)](#). Note that SEDS allows to only learn models where the input and output variables have the same dimensions. Since the variables of the distribution $\mathcal{P}(\Psi(\xi_x), \xi_f | \theta_{inf}^g)$ have not the same dimension, we learned this distribution through a variant of SEDS where we maximize the likelihood of the model under the constraint:

$$\mathbb{E}[\xi_f | \mathbf{0}] = \mathbf{0}. \quad (2.8)$$

REPRODUCTION

While reproducing the task, the model essentially works in three phases: *Update hand position* \rightarrow *Infer finger joints* \rightarrow *Increment finger joints*. The palm position is updated independently at every time step and its current value is used to modulate the dynamics of the finger motion through the coupling mechanism. [Figure 2.2](#) shows this flow of information across the sub-systems and the robot. Such a scheme is desired since it ensures that any perturbation is reflected appropriately in both sub-systems.

The process starts by generating a velocity command for the hand transport sub-system and increments its state by one time step. $\Psi(\xi_x)$ transforms its current state which is fed to the inference model that calculates the desired state of the finger joint angles by conditioning the learned joint distribution.

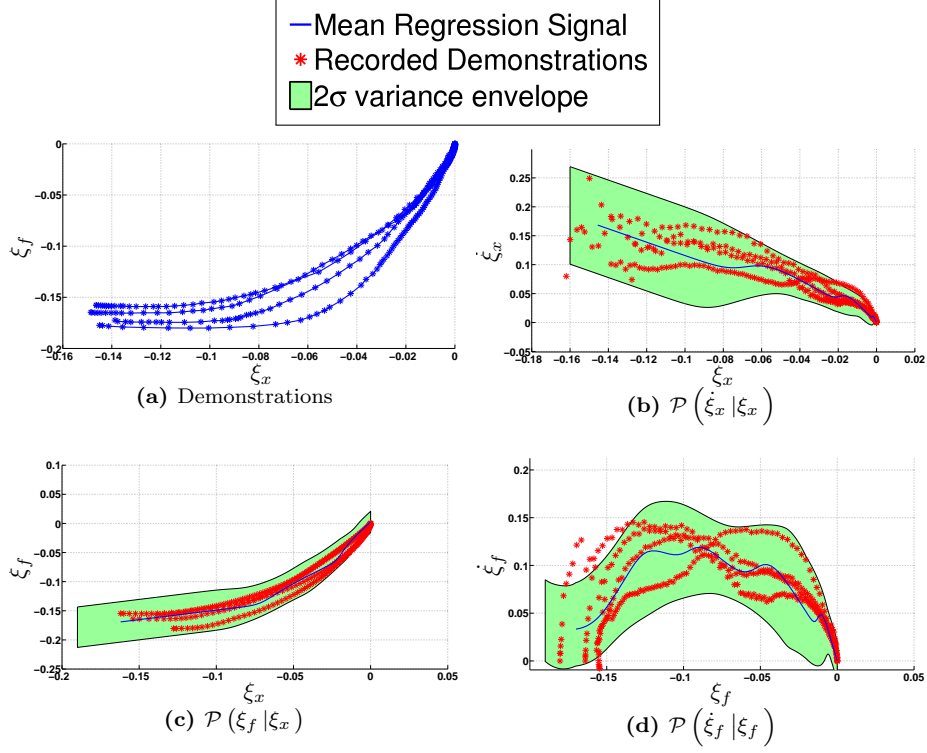


Figure 2.3: GMMs which combine to form the CDS model for 2D example. (a) shows the human demonstrations. Large number of datapoints around the end of trajectories depict very small velocities. (b) shows the GMM encoding the velocity distribution conditioned on the position of reaching motion (ξ_x), (c) shows the GMM encoding the desired value of ξ_f (i.e. $\tilde{\xi}_f$) given the current value of ξ_x as seen during the demonstrations. (d) shows the GMM encoding the dynamic model for the finger pre-shape.

The velocity to drive the finger joints from their current state to the inferred (desired) state is generated by *Gaussian Mixture Regression* (GMR) conditioned on the error between the two. The fingers reach a new state and the cycle is repeated until convergence. Algorithm 2.1 explains the complete reproduction process in pseudo-code.

Note that the coupling function $\Psi(\xi_x)$ also acts as a phase variable which updates itself at each time step and, in the event of a perturbation, will command the fingers to re-adjust so as to maintain the same correlations between the sub-system states as learned from the demonstrations. Two other parameters governing the coupled behavior are scalars $\alpha, \beta > 0$. Qualitatively speaking, they respectively control the speed and amplitude of the robot’s reaction under perturbations.

As described in Section 2.4.1, learning using SEDS ensures that the model for reaching and the model for grasping are both stable at their respective attractors. We however need now to verify that when one combines the two models using CDS, the resulting model is stable at the same attractors.

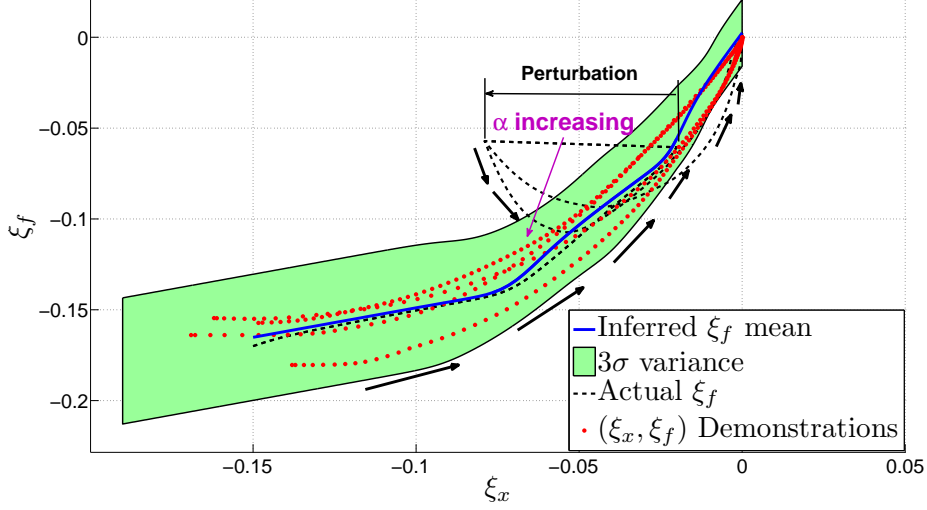


Figure 2.4: Reproducing the task under the CDS model. The reproduction (dashed) is overlaid on the demonstrations for reference. The model is run for different α values and the flow of the state values in time is depicted by the arrows. It is evident that the model tries to track the desired ξ_f (blue) values at the current ξ_x by reversing the velocity in ξ_f direction. The tracking is more stringent for larger α .

Algorithm 2.1 Task Execution using CDS

Input: $\xi_x(0); \xi_f(0); \theta_x^g; \theta_{inf}^g; \theta_f^g; \alpha; \beta; \Delta t; \epsilon$

Set $t = 0$

while $(\|\dot{\xi}_f(t)\| > \epsilon \text{ and } \|\dot{\xi}_x(t)\| > \epsilon)$ **do**

if perturbation **then**

 update $g \in \mathcal{G}$

end if

 Update Hand Position: $\dot{\xi}_x(t) = \mathbb{E} \left[\mathcal{P} \left(\dot{\xi}_x | \xi_x; \theta_x^g \right) \right]$

$\xi_x(t+1) = \xi_x(t) + \dot{\xi}_x(t) \Delta t$

 Infer Finger Joints: $\dot{\xi}_f(t) = \mathbb{E} \left[\mathcal{P} \left(\dot{\xi}_f | \Psi(\xi_x); \theta_{inf}^g \right) \right]$

 Update Finger Joints: $\dot{\xi}_f(t) = \mathbb{E} \left[\mathcal{P} \left(\dot{\xi}_f | \beta(\xi_f - \tilde{\xi}_f); \theta_f^g \right) \right]$

$\xi_f(t+1) = \xi_f(t) + \alpha \dot{\xi}_f(t) \Delta t$

$t \leftarrow t + 1$

end while

Definition. A CDS model is globally asymptotically stable at the attractors ξ_x^*, ξ_f^* if by starting from any given initial conditions $\xi_x(0), \xi_f(0)$ and coupling parameters $\alpha, \beta \in \mathbb{R}$ the following conditions hold:

$$\lim_{t \rightarrow \infty} \xi_x(t) = \xi_x^* \quad (2.9a)$$

$$\lim_{t \rightarrow \infty} \xi_f(t) = \xi_f^* \quad (2.9b)$$

Such a property is fundamental to ensure that the CDS model will result in a reach and grasp motion terminating at the desired target. Most importantly,

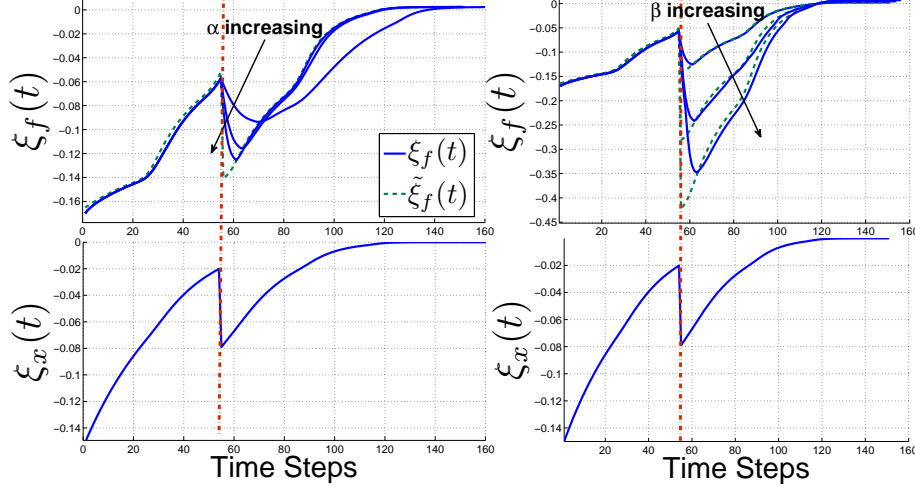


Figure 2.5: Variation of obtained trajectories with α and β . Vertical red line shows the instant of perturbation when the target is suddenly pushed away along positive ξ_x direction. Negative velocities are generated in ξ_f in order to track $\tilde{\xi}_f$. Speed of retracting is proportional to α (left) and amplitude is proportional to β (right).

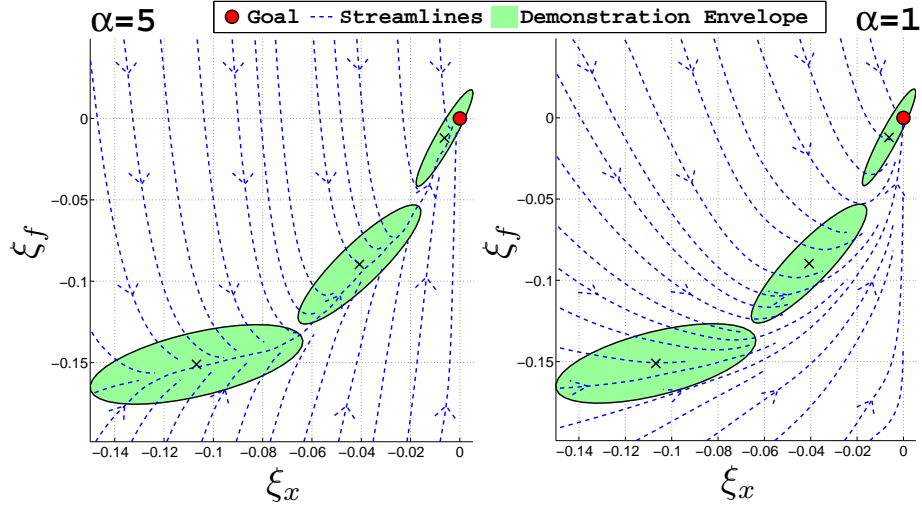


Figure 2.6: Change in α affecting the nature of streamlines. Larger α will tend to bring the system more quickly towards the (ξ_x, ξ_f) locations seen during demonstrations.

showing that the attractors for hand and fingers are also globally asymptotically stable will ensure that this model benefits from the same robustness to perturbation as described for the simple reaching model in [Section 2.4.1](#). See [Appendix A.1](#) for the proof of stability.

MINIMAL EXAMPLE

To establish an intuitive understanding, we instantiate the CDS model as a 2D representative example of actual high-dimensional reach-grasp tasks. We consider 1-D cartesian position ξ_x of the end-effector and 1 finger joint angle ξ_f , both expressed with respect to their respective goal states so that they

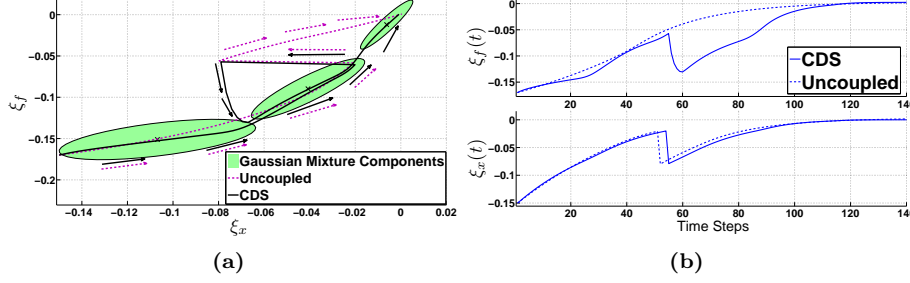


Figure 2.7: Task reproduction with explicit and implicit coupling shown in (a) state space, (b) time variation. Dotted lines show the implicitly coupled task execution. Note the difference in the directions from which the convergence occurs in the two cases. In the explicitly coupled execution, convergence is faster in ξ_x than in ξ_f .

converge to the origin. In this way, the full fledged grasping task is just a higher dimensional version of this case by considering 3-dimensional cartesian position instead of ξ_x and all joint angles (or eigen-grasps) of the fingers instead of ξ_f .

Under the given setting, typical demonstrations of reach-grasp task are as shown in Figure 2.3(a), where the reaching motion converges slightly faster than the finger curl. We extract the velocity information at each recorded point by finite differencing and build the following models from the resulting data: $\mathcal{P}(\xi_x, \dot{\xi}_x | \theta_x)$, $\mathcal{P}(\xi_x, \xi_f | \theta_{inf})$ and $\mathcal{P}(\xi_f, \dot{\xi}_f | \theta_f)$. The resulting mixtures for each of the models is shown in Figure 2.3. For reproducing the task, instead of using the earlier approach of Gribovskaya and Billard (2009) where the system evolves under the velocities computed as $\mathbb{E}[(\dot{\xi}_x; \dot{\xi}_f) | (\xi_x; \xi_f)]$, we proceed as in Algorithm 2.1. Figure 2.4 shows reproduction of the task in the (ξ_x, ξ_f) space overlaid on the demonstrations. It clearly shows that a perturbation in ξ_x creates an effect in ξ_f , i.e., generating a negative velocity, the magnitude of which is tunable using the α parameter. This change is brought due to the need of tracking the inferred ξ_f values i.e. $\tilde{\xi}_f$, at all ξ_x . $\tilde{\xi}_f$ represents the expected value of ξ_f given ξ_x as seen during the demonstrations. The variation of the trajectories of ξ_f with α and β is shown in Figure 2.5. α modulates the speed with which the reaction to perturbation occurs. On the other hand, a high value of β increases the amplitude of reopening. Figure 2.6 shows the streamlines of this system for two different α values in order to visualize the global behavior of trajectories evolving under the CDS model.

At this point, it is important to distinguish our approach from the single GMM approach of Gribovskaya and Billard (2009) mentioned in Section 2.4.2. Figure 2.7 shows a comparison of the CDS trajectories with those obtained using the single GMM approach, where the coupling is only implicit. It shows the behavior when a perturbation is introduced only on the abscissa. Clearly, in the implicitly coupled case, the perturbation is not appropriately transferred to the unperturbed dimension ξ_f and the motion in that space remains unchanged. This behavior can be significantly different depending on the state of the two

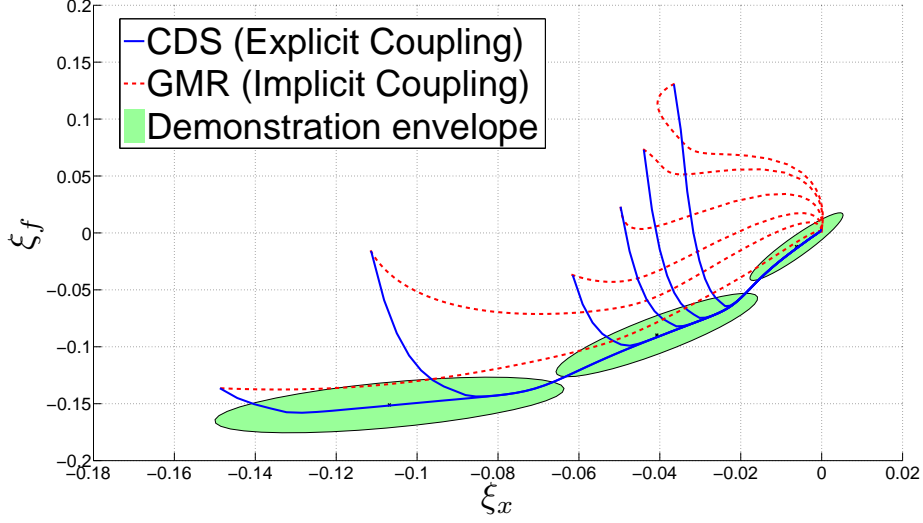


Figure 2.8: Comparison between the motions obtained by single GMM and CDS approaches. Note how the order of convergence can be remarkably different when starting at different positions in state space.

sub-systems just after the perturbation.

To investigate this, we initialize the single GMM model as well as the CDS model at different points in state space and follow the two trajectories. Figure 2.8 shows this experiment. Notice the sharp difference in the trajectories as the CDS trajectories try to maintain correlation between the state space variables and always converge from within the demonstration envelope. On the other hand, the trajectories of the single GMM approach have no definite convergence constraint². This difference is significantly important in the context of reach-grasp tasks. If the trajectories converge from the top of the envelope, it means that the variable ξ_f (fingers) is converging faster than the ξ_x (hand position). This translates to premature finger closure as compared to what was seen during the demonstrations. If they converge from below, it means that the fingers are closing later than what was seen during the demonstrations. While the former is undesirable in any reach-grasp task, the latter is undesirable only in the case of moving/falling objects.

2.5 Experiments and Results

A core assumption of our approach lies in the fact that human motor control exploits an explicit coupling between hand and finger motions. In this section, we first validate this hypothesis by reporting on a simple motion studies conducted with five subjects performing a reach and grasp task under perturbations. Data from human motion are used in three capacities: a) to confirm that the

²It is also worth mentioning that the two trajectories are fairly similar when initialized close to the demonstration envelope.

CDS model captures well the coupling across hand and fingers found in human data; b) as demonstration data to build the probability density functions of the CDS model; c) to identify relationships across the variables of the system and to use these to instantiate the two free parameters (α and β) of the CDS model.

In the second part of this section, we perform various experiments in simulation and with the real iCub robot to validate the performance of the CDS model as a good model to ensure robust control of reach and grasp in robots. In particular, we test that the CDS model is indeed well suited to handle fast perturbations which typically need re-planning and are difficult to handle on-line. Videos for all robot experiments and simulations are cross-linked to the corresponding figures.

2.5.1 INSTANTIATION OF CDS VARIABLES

In all the experiments presented here, including human data, the state of our system is composed of the cartesian position and orientation of the end-effector (human/robot wrist) and of the following 6 finger joint angles:

- 1 for curl of the thumb.
- 2 for index finger proximal and distal joints.
- 2 for middle finger proximal and distal joints.
- 1 for combined curl of ring and little finger.

We use the norm-2 for the coupling function, i.e. $\Psi(.) = ||.||$ in the CDS implementation for modeling both human data and for robot control. As a result, fingers' reopen and close as a function of the distance of the hand to the target.

Since CDS controls for the hand displacement, we use the moore-penrose inverse kinematic function to convert the end-effector pose to joint angles of the arm. In simulation and on the real iCub robot, we control the 7 degrees of freedom (DOFs) of the arm and 6 finger joints at an update rate of 20 ms.

2.5.2 VALIDATION AGAINST HUMAN DATA

As discussed in [Section 2.3.2](#), many physiological studies reported a natural coordination between arm and fingers when humans reach for objects. In order to assess quantitatively these observations and provide data in support of our model of a coupling between the two processes of hand transport and finger motion, we performed experiments with human subjects performing reach-to-grasp tasks under *fast and random perturbations*.

EXPERIMENTAL PROCEDURE

The experimental setup is shown in [Figure 2.1\(a\)](#). The subject stands in front of two stationary targets, a green and a red ball. An on-screen target selector prompts the subject to reach and grasp one of the two balls depending on the color shown on the screen. To start the experiment, one of the ball is switched on and the subject starts to reach towards the corresponding object. As the subject is moving his hand and preshaping his fingers to reach for the target ball, a perturbation is created by abruptly switching off the target ball and lighting up the second ball. The switch across targets occurs only once during each trial about 1 to 1.5 sec. after the onset of the motion. The subject’s hand has usually by then traveled more than half the distance separating it from the target. The trial stops once the subject has successfully grasped the second target.

To ensure that we are observing natural response to such perturbations, subjects were instructed to proceed at their own pace and no timing for the overall motion was enforced. As a result, the time it took for each subject to complete the motion varied across subjects and across trials. Since encoding in the CDS model is time-invariant, modeling is not affected by these changes in duration of experiment completion.

We recorded the kinematic of the hand, fingers and arm motions of 5 subjects across 20 trials. 10 of the trials were unperturbed, i.e. the target was not switched during the motion. Subjects did the 20 trials in one swipe. Unperturbed and perturbed trials were presented in random order for each subject. The arm and hand motion was recorded using three XSensTM IMU motion sensors attached to the upper arm, forearm and wrist of the subject at a frame rate of 20ms³. The fingers’ motion was recorded using a 5DTTM data glove. Angular displacements of the arm joint and finger joints were re-constructed and mapped to the iCub’s arm joint angles and finger joint angles. To assess visually that the correspondence between human motion and robot motion is well done, the iCub simulator runs simultaneously while the human is performing the trials.

Data from the 10 unperturbed trials and from the 5 subjects are used to train the 3 GMM-s which serve as basis for the CDS model. The next section discusses how well the CDS model renders human behavior under perturbations.

QUALITATIVE ANALYSIS OF HUMAN MOTION

Visual inspection of the human data confirms a steady coupling between hand transport and fingers closing in the unperturbed situation, whereby fingers close faster as the hand approaches faster the target, and conversely. This coupling persists across trials and for all subjects. Most interesting is the obser-

³To compensate for drifts from the IMU and data glove measurements, subjects were instructed to proceed to a brief calibration procedure after each trial. This procedure lasted no more than 5 sec.

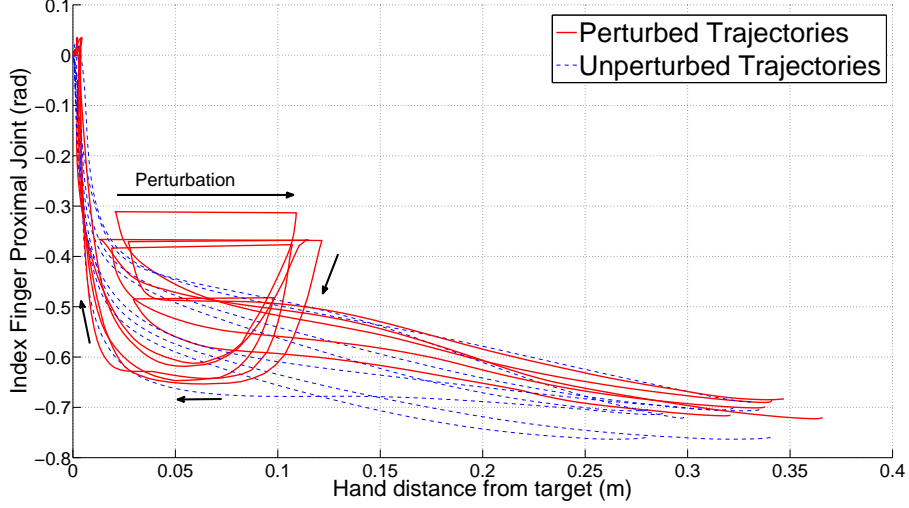


Figure 2.9: Hand-finger coordination during perturbed and unperturbed demonstrations.

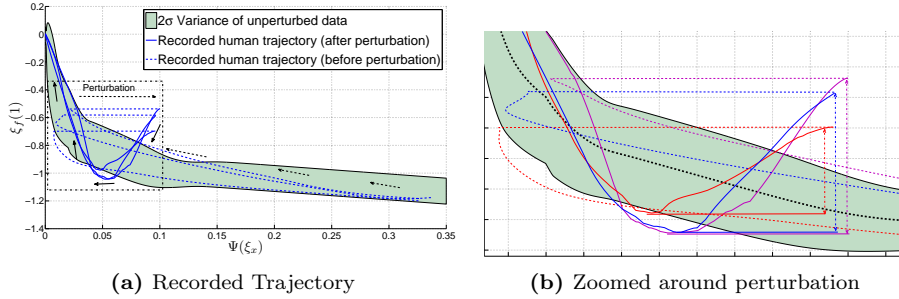


Figure 2.10: (a) shows the data recorded from perturbed human demonstrations. The adaptation behavior under perturbation follows the same correlations between hand position and fingers as in the unperturbed behavior. The region where perturbation is handled is indicated in red and zoomed in (b) where 3 different demonstrations (red, blue and magenta) from the same subject are shown.

vation that, during *perturbed* trials, just after the target is switched, the fingers first reopen and then close again synchronously with the hand, as the hand moves toward the new target, see Figure 2.1(b). Note that, in all trials, the fingers re-opened irrespective of the fact that the aperture of the fingers at the time of perturbation was large enough to accommodate the object. This suggests that this reaction to perturbation is not driven by the need of accommodating the object within the grasp, but may be the result of some inherent property of finger-hand motor control. It appears as if the fingers would first “reset” to a location that corresponds to the expected location for the fingers given the new hand-target distance. Once reset, fingers and hand would resume their usual coupled hand-finger dynamics. Figure 2.9 shows this typical two-phase motions after perturbation, plotting the displacement of the proximal joint of the index finger against the distance of the hand to the target ball.

The CDS model, using the distance of the hand to the target for the coupling function $\Psi(\xi_x)$, gives a very good account of this two phases response and is

shown overlaid on the data, see [Figure 2.10\(a\)](#). Observe further that the trajectories followed by the fingers after perturbations remain within the covariance envelope of the model. This envelope represents the variability of finger motion observed during the *unperturbed* trials. This hence confirms the hypothesis that the fingers resume their unperturbed motion model shortly after responding to the perturbation. This is particularly visible when looking at [Figure 2.10\(b\)](#), zoomed in on the part of the trajectories during and just after the perturbation. Three different demonstrations are shown. It can be seen that, irrespective of the state of the fingers ξ_f at the time of perturbation, the finger trajectories tend to follow the mean of the regressive model (which is representative of the mean of the trajectories followed by the human finger during the unperturbed trials) before the perturbation occurs. Just after the perturbations, the fingers then re-open (trajectory goes down) and then close again (trajectory goes up).

MODELING HUMAN MOTION

The previous discussion assessed the fact that the CDS model gives a good account of the qualitative behavior of the fingers’s motion after perturbation. We here discuss how, by tuning the two open parameters of the CDS models, namely α and β (see [Table 2.1](#)), we can better reproduce individual trajectories of the fingers for a particular trial and subject.

As illustrated in [Section 2.4.3](#), these two parameters control, respectively, for the speed and amplitude of the motion of the reopening of the fingers after perturbation. Although these parameters can be set arbitrarily in our model, a closer analysis of human data during the perturbed trials shows that one can estimate these parameters by observing the evolution of hand motion *prior* to perturbation. When plotting the average velocity of the hand prior to perturbation and the amplitude of finger reopening, we see that the two parameters are linearly correlated, see [Figure 2.11\(a\)](#). Similarly, when plotting the velocity at which fingers reopen against the amplitude of reopening, we see that these two parameters are also linearly correlated. In other words, the faster the hand moves towards the target, the less the fingers reopen upon perturbation. Further, the faster the fingers reopen the larger the amplitude of the reopening of the fingers. Note that while there is a correlation, this correlation is subject dependent. To reproduce human data for a particular trial with CDS, we can hence use the above two observations combined with the fact that α and β control the speed and amplitude of the fingers’ motion.

[Figure 2.11\(c\)](#) shows that this results in a good qualitative fit of the motion after perturbation. 3 perturbed trials chosen from subject 1 are shown. Similar plots for other subjects can be found in [A.1.1](#). We analyze the quality of the fit by comparing it to the motion obtained from the model with *optimal* parameter values. We find the optimal values of α and β for a particular demonstration by performing a grid search and optimizing the fit between the model generated and

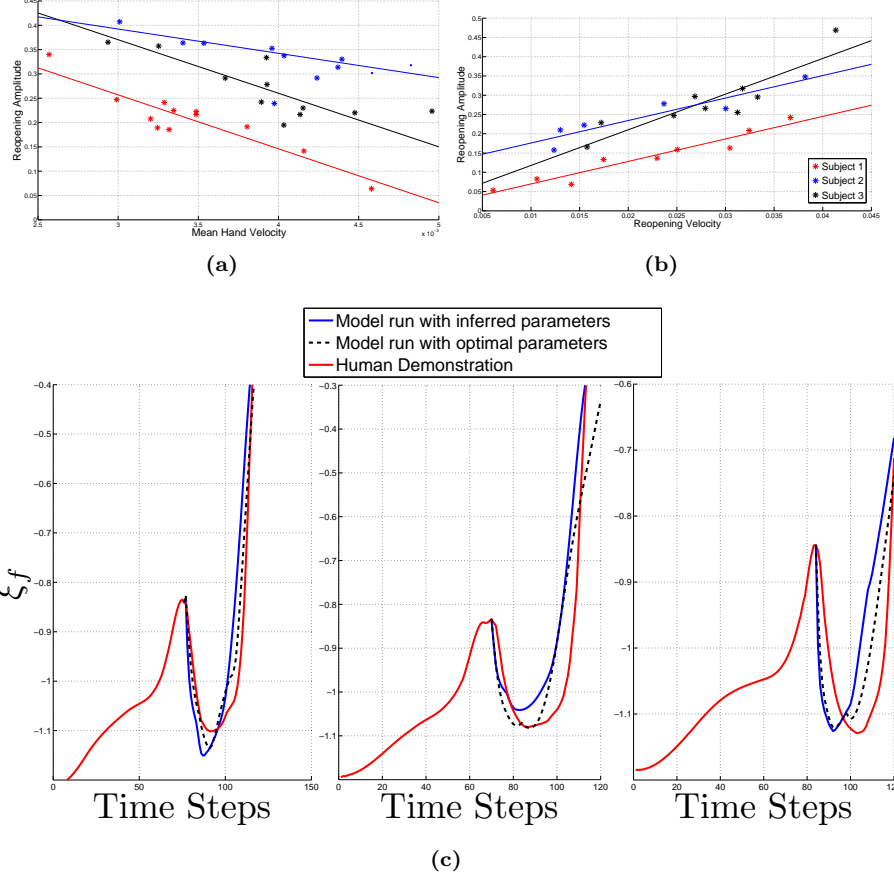


Figure 2.11: Correlations deduced from the experiments. (a) shows the linear correlation found between mean hand velocity prior to perturbation and the amplitude of finger reopening. (b) shows the same between speed and amplitude of finger reopening. (c) compares the trajectories obtained using the inferred parameters with the actual human demonstration. Finger motion obtained using optimal parameters values is also shown in black (dashed).

demonstrated motion⁴. The fitting is evaluated using the absolute error between the joint angle values (radians) summed over a time window from the instant of perturbation till the end of demonstration. Figure 2.12 shows the variation of this error term with α and β . It can be seen that the error first decreases and then increases with progressively increasing α and β . The contours of the error function on variation with α and β are shown in Figure 2.12(c)

Note that it is not the aim of this analysis to find the optimal parameters, but to give the reader an idea of how good is the fit obtained from a brute force grid search as compared to what we can infer *prior* to the perturbation. Further, the discrepancy between the model run with inferred parameters and the actual data is only due to the noise in the linear correlation.

It is important to emphasize that the CDS model is built using data from the unperturbed trials and the parameters α and β are inferred from the perturbed trials. It is a representative of a generic pattern of finger-hand coupled dynamics

⁴This estimate of the *optimal* α and β is only accurate upto the width of the grid chosen.

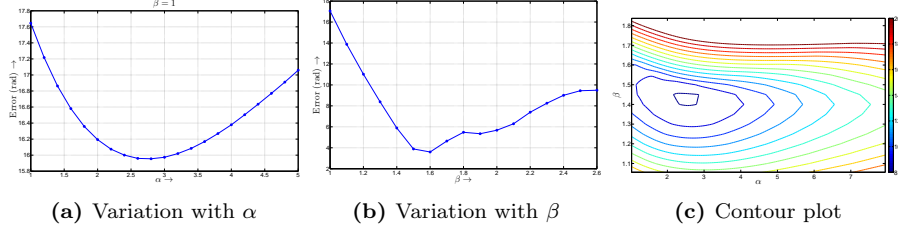


Figure 2.12: Effect of changing parameters α and β on the error between model run and mean human demonstrations.

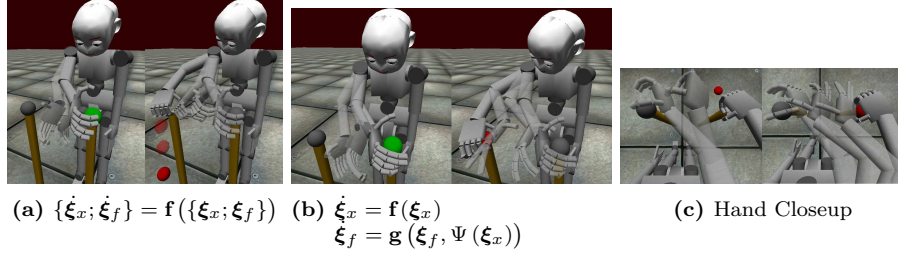


Figure 2.13: Reach-grasp task executions with and without explicit coupling. The explicitly coupled execution (b) prevents premature finger closure, ensuring that given any amount of perturbation, formation of the grasp is prevented until it is safe to do so. In the implicitly coupled execution (a), fingers close early and the grasp fails. (c) shows closeup of hand motion post perturbation with implicit (left) and explicit (right) coupling.

that is present across subjects and trials but that is not subject specific. Further, the estimation of the parameters α and β is done based on an observation of a coupling across variables in a single subject, and is not fitted for a particular trial. The CDS model, hence, encapsulates general patterns of finger-hand motions inherent to human motor control. We discuss next how such human-like dynamics of motion can be used for robust control of hand-finger motion for successful grasp during perturbations.

2.5.3 VALIDATION OF THE MODEL FOR ROBOT CONTROL

We here test the performance of CDS for robust control of reach and grasp motion in the iCub robot. We first show using the iCub simulator that the approach presented in this work is decidedly better than our previous approach of learning task dynamics using only one dynamical system. It ensures successful task completion under spatial perturbation of the target where the previous approach fails. We also investigate the adaptability of the CDS model in reacting quickly to counter fast perturbations (even when not demonstrated a priori). Finally, we conduct experiments on the iCub robot to validate the ability of the model to adapt on-the-fly reach and grasp motion under various forms of perturbations.

In [Section 2.4.3](#) we discussed the fact that the “naive” approach in which one would learn the hand-finger coupling by using a single GMM (the state vector in this case comprises the hand position and finger joints) would likely fail at encapsulating explicitly the correlation between the two dynamics. We had then advocated the use of an explicit coupling function to couple the dynamics of finger and hand motion, each of which are learned through separate GMM-s, leading to the CDS model. We here illustrate this in simulation when reproducing the human experiment. The iCub robot first reaches out for the green ball. Midway through the motion, the target is switched and the robot must go and reach for the red ball.

[Figure 2.13\(a\)](#) shows that using a single GMM for the hand and finger dynamics fails at embedding properly the correlations between the reach and grasp sub-systems and does not adapt well the fingers’ motion to grasp for the new ball target. Lack of an explicit coupling leads to a poor coordination between fingers and hand motion. As a result, the fingers close too early, leading the ball to fall. [Figure 2.13\(b\)](#) shows the same task when performed using the CDS model. The fingers first reopen following the perturbation, hence delaying the grasp formation, and then close according to the correlations learned during the demonstrations, leading to a successful grasp. [Figure 2.13\(c\)](#) shows the hand from top view where the re-opening of fingers can be seen clearly in the explicitly coupled task.

ADAPTABILITY TO FAST PERTURBATIONS

An important aspect of encoding motion using autonomous dynamical systems is that it offers a great resilience to perturbations. We here show that this offers robust control in the face of very rapid perturbations.

In [Section 2.5.2](#), we showed that the two free parameters α and β of the CDS model could be inferred from human data. We then already emphasized the role of these two parameters to control for speed and amplitude of finger reopening. To recall, the larger α the faster the motion. Hence, executing the task with values for α that differ from that set from human data may be interesting for robot control for two reasons: a) as robots can move much faster than humans, using larger values for α could exploit the robot’s faster reaction times while retaining the coupling between finger and hand motion found in human data. b) Also, using values of α that depart from these inferred from human demonstration may allow to generate better responses to perturbations that send the system to area of the state space not seen during demonstrations.

[Figure 2.14\(a\)](#) illustrates the role that α plays in controlling for the reaction time. As expected, the time it takes for the finger to adapt to perturbation

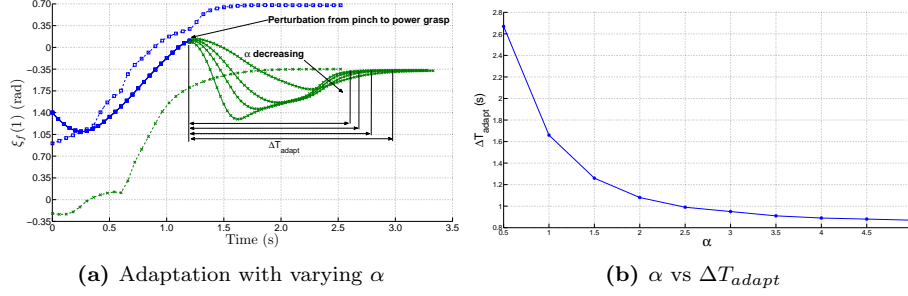


Figure 2.14: (a) - Speed of reaction varies with varying α . Adaptation is qualitatively the same, but faster as α increases. (b) shows the variation of recovery time ΔT_{adapt} with α .

decreases when increasing α (i.e. ΔT_{adapt}) corresponds to the time elapsed between the onset of the perturbation and the time when the finger position rejoin the original desired position, i.e. the position the finger should have been in the unperturbed case. Figure 2.14(b) plots α against ΔT_{adapt} . Recovery times can be significantly reduced by increasing α . This is the time it takes for the robot to completely recover from the perturbation and reach the target position successfully. It is important to emphasize once more that this trajectory “re-planning” is performed at run-time, i.e within the 20 ms close-loop control of the robot. Again, there is no replanning, adaptation to perturbation results from providing the CDS model with the current position of the finger, hand and target. Importantly, this provides a smooth response that enables the robot to change its trajectory *without stopping to re-plan*.

We illustrate this capacity to adapt to rapid perturbation in an experiment with the iCub robot when the robot must not only adapt the trajectory of its hand but also switch across grasp types, see Figure 2.15. Due to hardware constraints on the real platform, we perform this particularly high speed perturbation experiment in the iCub simulator. As the robot moves towards the target located on its left, the target ball suddenly disappears and reappears on the right of the robot. In contrast to our previous experiment, the ball is no longer supported against gravity and hence, starts falling. For the robot to reach and grasp the object before it reaches the floor, the robot has to act very quickly. This requires a fast adaptation from palm-up to palm-down grasp as well as for fingers, while the target keeps on moving. To perform this task, we first trained two separate CDS model to learn two different dynamics for power grasp in palm-up and palm-down configurations, respectively. Learning was done by using five (unperturbed) human demonstrations of this task. During reproduction, the robot initially starts moving towards the target using the CDS model for palm-down configuration grasp. After perturbation, the robot switches to the CDS model for palm-up power grasp.

To ensure that the robot intercepts the falling object in its workspace, we use the approach presented in our previous work on catching flying objects by

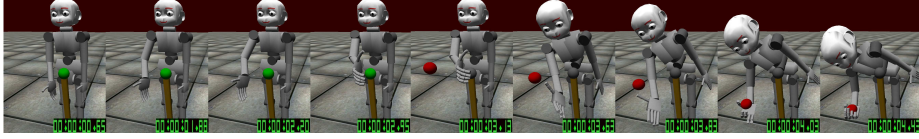


Figure 2.15: Fast adaptation under perturbation from palm-down to palm-up power grasp.

T_p (s)	ΔT_{adapt} (s)
1.2	2.82
1.5	2.77
1.8	2.76
2.1	2.80

Table 2.1: Variation of time taken to recover from perturbation with the instant of perturbation. Note that the total task duration is of the order of 4 s. The values were taken at constant α and hence do not change with the instant of perturbation. This shows the robustness of the proposed method in adapting against perturbations.

Kim et al. (2010)⁵. This allows us to determine the catching point as well as the time it will take the object to reach this point (assuming here a simple free fall for the dynamics of the object). This determines the maximal value for ΔT_{adapt} , which we then use to set the required α to intercept the object in time.

As shown in Figure 2.15, switching to the second CDS model ensures that re-planning of the finger motion is done in coordination with the hand motion (now redirected to the falling object). Precisely, the orientation of the hand and the finger curl are changed synchronously yielding the hand to close its grasp on the falling object at the right time. Notice that as the distance-to-target suddenly increases, the CDS model forces the fingers to reopen. Subsequently, the fingers close proportionally as the distance between the falling ball and the robot hand decreases, hence maintaining the correlations seen during the demonstrations. Note that to generate this task, we control also for the torso (adding two more variables to the inverse kinematics so as to increase the workspace of the robot).

SWITCHING BETWEEN DIFFERENT GRASP TYPES

Here, we perform another experiment showing the ability of our system to adapt the *fingers' configuration* (in addition to adapting the fingers' dynamics of motion) so as to switch between pinch and power grasps. We learn two separate CDS models for pinch grasp of a thin object (screw-driver) and power grasp of a spherical object, respectively, from five demonstrations of each task during unperturbed trials.

Figure 2.16 illustrates the experiment. While the robot reaches for the thin object, pre-shaping its fingers to the learned pinch grasp, we suddenly present the spherical object in the robot's field of view. The robot then redirects its hand

⁵In Kim et al. (2010) we had used a single GMM to control for both arm and hand motion. This would not allow to quickly adapt to different grasp on the fly as shown in Section 2.5.3

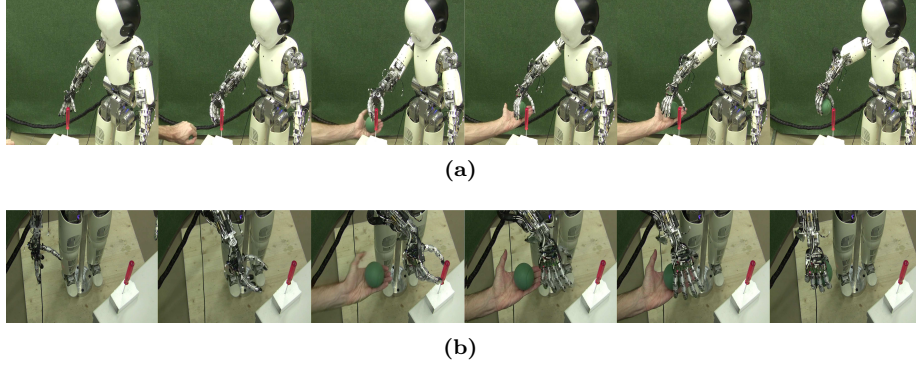


Figure 2.16: Validating the model on the real iCub platform. The robot adapts between pinch and power grasps at different spatial positions in real time without any delays for re-planning. (a) and (b) show the same task from front and top view to better visualize the motion of the fingers.

to reach for the spherical object in place of the thin one. Since this experiment does not require very rapid reacting time, the experiment could be conducted on the real iCub robot. In this experiment, the two objects are color-tracked using the iCub’s on-board cameras. Change of target is hardcoded. As soon as the green object is detected in the cameras, the target location is switched from the red object to the green one and the robot’s CDS model is switched accordingly.

Figure 2.17 shows the motion of robot’s index finger proximal joint as it adapts to the induced perturbation. During the first phase of the motion, the finger closes rapidly so as to yield a pinch grasp. After perturbation, the fingers reopen to yield the power grasp that would better accommodate the spherical object. The robot smoothly switches from following the pinch-grasp model requiring smaller hand aperture (i.e. larger joint value) to the power grasp model which requires a larger aperture (smaller joint value) by reopening the fingers and subsequently closing them on the target, thereby, completing the task successfully.

While we discussed in the previous section the advantage to adapt the parameter α when one needs to perform tasks that require very high reaction times (reaction times that are higher than what humans could achieve), we here show that, using the α parameter inferred from human data is sufficient when required reaction times are sufficiently slow. We emphasize once more the benefit of the model to achieve very robust behavior in the face of various perturbations. To this end, we perform two variants on the task described above where we introduce perturbations.

First, we run the same switching tasks but present the spherical objects at different instants after the onset of motion. The perturbation instants vary from middle of the task duration to almost completion of the task. Table 2.1 gives the recovery time and time instant of perturbation. T_p denotes the instant at which the perturbation was introduced. Since we do not change the value of

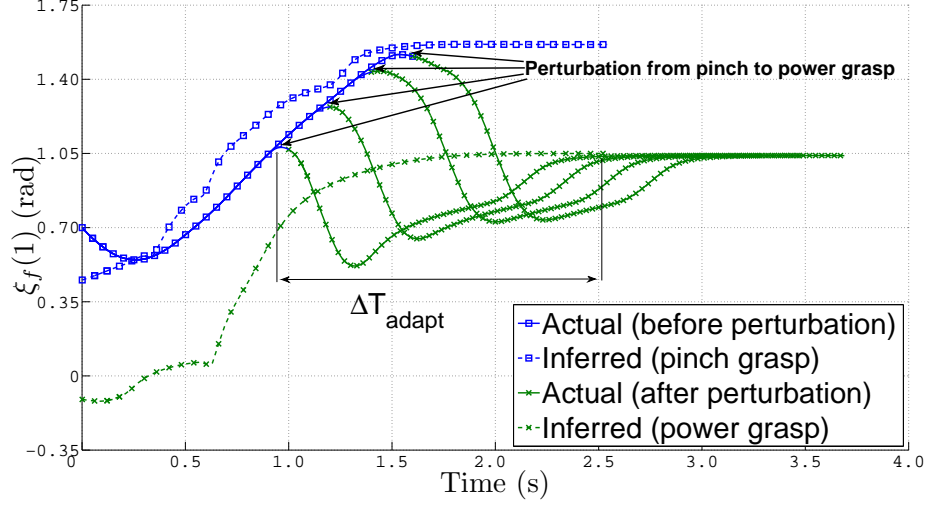


Figure 2.17: Motion of one finger joint angle under grasp-type perturbation as recorded from the simulation. The inferred joint position predicted by the models both the grasp models (power and pinch) are shown in dotted. The adaptation is smooth and robust w.r.t the instant when perturbation was applied. Time taken to recover from perturbation (ΔT_{adapt}) remains constant.

the parameter α , at each run, the time taken by the robot to recover from the perturbation remains the same. Figure 2.17 shows the resulting trajectories for the index finger. Even when the perturbation occurs shortly before completion of pinch grasp, the model readapts the grasp smoothly, yielding a correct grasp at the second object.

Second, to highlight the performance of CDS to adapt continuously and on the fly control for coordinated motion of hand and fingers on the real iCub robot, we introduce perturbation during the first part of the previous task in which the iCub robot reaches with a pinch grasp (here the robot reaches for a glass of wine⁶), see Figure 2.18. To introduce perturbations on-the-fly during execution of the pinch grasp, we implement a reflex behavior using the iCub’s skin touch sensors on the forearm, such that, when the robot detects a touch on its forearm, it immediately moves its arm away from the point where it was touched. This reflex overlays the CDS controller. When the CDS controller takes over again, it uses the new position of the arm to predict the new finger and hand motion.

Figure 2.18 shows the displacement along time of the proximal and distal finger joints of the index finger, hand aperture and the distance-to-target, when the robot is solely reaching with a pinch grasp and is being perturbed once on its way toward the object. The hand aperture is computed as the distance between the tips of the thumb and index fingers. As expected, as the hand is moved away from the target, the fingers reopen in agreement with the correlations learned in the CDS model and then close into pinch grasp on the object. The

⁶The location of the grasping point on the glass of wine is indicated by a red patch that is detected through two external cameras running at 100Hz.

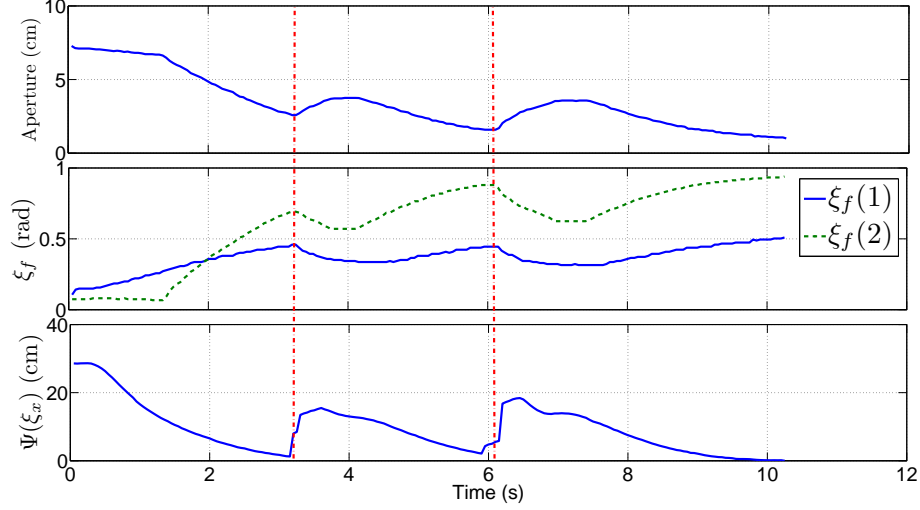


Figure 2.18: Coordinated hand arm motion while the robot was perturbed multiple times in different directions. Finger joint angles, and hence, the hand aperture, change according to the learned correlations. Vertical red line marks the instants at which the robot was perturbed.

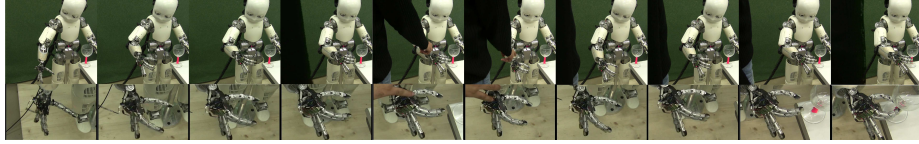


Figure 2.19: Spatio-temporal perturbations created using the tactile interface of the iCub. Top row shows the motion of the robot. Bottom row shows closeup of the hand.

finger motion robustly adapts to the perturbation, changing the hand aperture in coordination with the perturbed hand position and finally reaches the target state for the pinch grasp with approximately 1 cm hand aperture. [Figure 2.19](#) shows one cycle of tactile perturbation on the real robot.

2.6 Conclusions

In this chapter we addressed one of the challenges in robust modelling of reach-to-grasp motions: coupling between the reaching and grasping components. We presented a model for encoding and reproducing different reach-to-grasp motions that allows to handle fast perturbations in real-time. We showed that this capacity to adapt without re-planning could be used to allow a smooth switching across different grasps. The model was strongly inspired from the way humans adapt reach and grasp motion under perturbation. Human data was used to determine a generic coupling between control of hand transport and finger aperture. It was also used to determine quantitative values for this coupling.

We first showed that the model gives a good qualitative account of human reach and grasp motions during perturbation. We then showed through both

simulation and real robot experiments that the CDS model provides a robust controller for a variety of reach and grasp motions in robots. Importantly, we showed that while human behaviour is a good source of inspiration, one can depart from this model by tuning the parameters of the model, to induce better robot performance. This follows a trend in programming by demonstration that emphasizes the fact that what is good for the human is not necessarily good for the robot (biological inspiration should be taken with a grain of salt).

In this chapter we assumed that there exists only one way in which a particular object can be grasped and that this corresponds to the demonstrations from which the model was learned. This assumption makes it difficult to complete the grasp if the object can be grasped in multiple ways and is presented in a pose different from the one seen during the demonstrations. By construction, the CDS model allows to have only a single attractor point and hence it is constrained to yield a single grasp pose. However, most objects can be grasped at different points on their graspable surface. In the next chapter, we investigate how dynamical system based models could be extended to learn not just a single attractor point but multiple attractors, each corresponding to a different grasp afforded by an object.

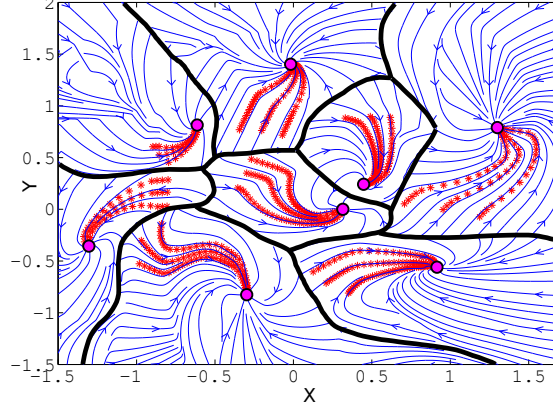
MULTIPLE ATTRACTOR DYNAMICS FOR REACH-TO-GRASP MOTIONS

3.1 Foreword

In previous chapters we have advocated the use of dynamical systems (DS) for encoding and generating reach-to-grasp motions. We showed that a major advantage of representing motion using DS based models (Pastor et al., 2009; Schöner and Dose, 1992; Schöner et al., 1995; Ellekilde and Christensen, 2009) is the ability to counter perturbations by virtue of the fact that re-planning of trajectories is instantaneous. However, so far in this thesis, and in many other related approaches (Reimann et al., 2011; Dixon and Khosla, 2004; Khansari Zadeh and Billard, 2011a) DS have been modelled with single point attractors. In the context of reach-to-grasp motions, this corresponds to the assumption of a single grasping location on the target object. Such an assumption constrains considerably the applicability of these methods to realistic scenarios. On the other hand, a DS composed of multiple stable attractors provides an opportunity to encode different ways to reach and grasp an object. Recent neuro-physiological results (Hoffmann, 2011) have shown that a DS based modelling best explains the trajectories followed by humans while switching between several reaching targets. From a robotics viewpoint, a robot controlled using a DS with multiple attractors would be able to switch online across different grasping strategies. This may be useful, e.g., when one grasping point becomes no longer accessible due to a sudden change in the orientation of the object or the appearance of an obstacle along the current trajectory. In this chapter we present a framework using which one can combine - in a single dynamical system - multiple dynamics directed toward different attractors. The work of this chapter lead to the following publications:

- A. Shukla and A. Billard. Augmented-SVM: Automatic space partitioning for combining multiple non-linear dynamics. In *Neural Information Processing Systems (NIPS) 25*, pages 1025–1033, 2012b
- A. Shukla and A. Billard. Augmented-svm for gradient observations with application to learning multiple-attractor dynamics. In *Support Vector Machines Applications, Chapter 1*, pages 1–21. Springer International Publishing, 2014. ISBN 978-3-319-02299-4. doi: 10.1007/978-3-319-02300-7_1

Figure 3.1 A DS with 8 attractors learned using only a few representative trajectories (red asterisks) from each DS. The bifurcation boundaries, as well as the dynamics of each DS need to be estimated from these trajectories.



3.2 Introduction

A non-linear dynamical system represented as $\dot{\mathbf{x}} = f(\mathbf{x})$ is usually estimated using non-linear regression approaches such as Gaussian Process Regression (GPR) (Rasmussen, 2004), Gaussian Mixture Regression (GMR) (Khansari Zadeh and Billard, 2011a), Locally Weighted Projection Regression (LWPR) (Schaal et al., 2002). However, all of these works modelled DS with a single attractor. While Khansari Zadeh and Billard (2011a); Pastor et al. (2009) ensure global stability at the attractor, other approaches result in unstable DS with spurious attractors.

Stability at multiple targets has been addressed to date largely through neural networks approaches. The Hopfield network and variants offered a powerful means to encode several stable attractors in the same system to provide a form of content-addressable memory (Fuchs and Haken, 1988; Michel and Farrell, 1990). The dynamics to reach these attractors was however not controlled for, nor was the partitioning of the state space that would send the trajectories to each attractor. Echo-state networks provide alternative ways to encode various complex dynamics (Jaeger et al., 2007). Although they have proved to be universal estimators, their ability to generalize in untrained regions of state space remains unverified. To our knowledge, the approach presented in this chapter is the first attempt at learning simultaneously a partitioning of the state space and an embedding of multiple dynamical systems with separate regions of attractions and distinct attractors.

3.3 Identifying dynamic constraints

A naive approach to building a multi-attractor DS would be to first partition the space and then learn a DS in each partition separately. This would unfortunately rarely result in the desired compound system. Consider, for instance, two DS with distinct attractors, as shown in Figure 3.2(a)-(b). First, we build a SVM classifier to separate data points of the first DS, labeled +1, from data points

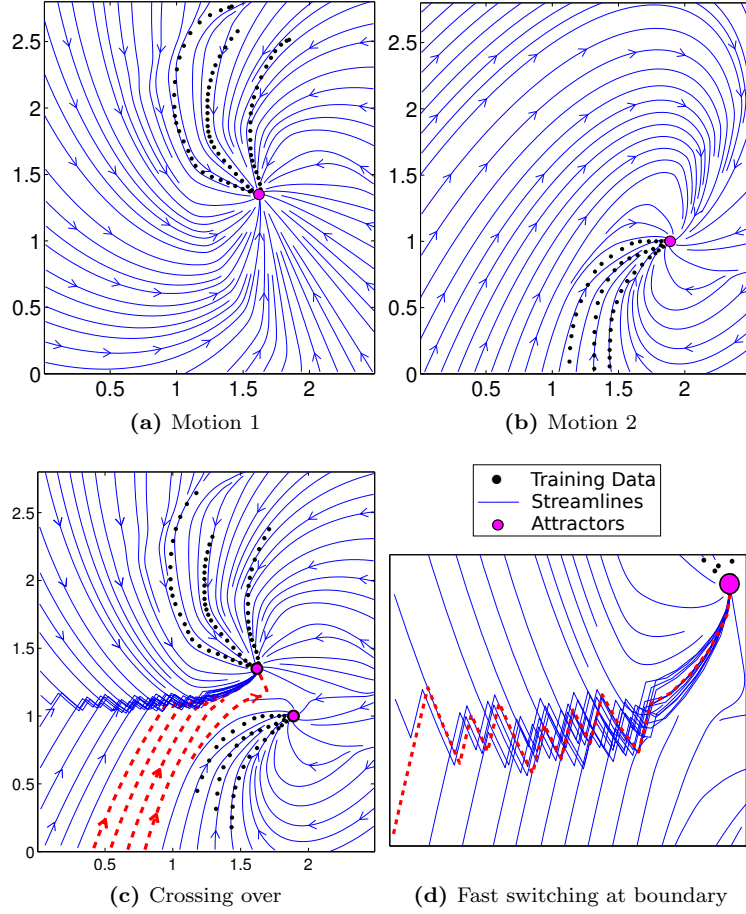


Figure 3.2: Combining motions using naive SVM classification based switching. (a), (b) - Two different dynamics with distinct attractors which are to be combined. (c) - Employing a simple switching scheme leads to crossing over of some trajectories shown in red. (d) - Zoomed in around the boundary, showing the fast switching near the boundary.

of the other DS, labeled -1 . We then estimate each DS separately using any of the techniques reviewed in the previous section. Let $h : \mathbb{R}^N \mapsto \mathbb{R}$ denote the classifier function that separates the state space $\mathbf{x} \in \mathbb{R}^N$ into two regions with labels $y_i \in \{+1, -1\}$. Also, let the two DS be $\dot{\mathbf{x}} = f_{y_i}(\mathbf{x})$ with stable attractors at $\mathbf{x}_{y_i}^*$. The combined DS is then given by $\dot{\mathbf{x}} = f_{\text{sgn}(h(\mathbf{x}))}(\mathbf{x})$. Figure 3.2(c) shows the trajectories resulting from this approach. Due to the non-linearity of the dynamics, trajectories initialized in one region cross the boundary and converge to the attractor located in the opposite region. In other words, each region partitioned by the SVM hyperplane is not a region of attraction for its attractor. In a real-world scenario where the attractors represent grasping points on an object and the trajectories are to be followed by robots, crossing over may take the trajectories towards kinematically unreachable regions. Also, as shown in Figure 3.2(d), trajectories that encounter the boundary may switch rapidly between different dynamics leading to jittery motion.

To ensure that the trajectories do not cross the boundary and remain within the region of attraction of their respective attractors, one could adopt a more informed approach in which each of the original DS is modulated such that the generated trajectories always move away from the classifier boundary. Recall that by construction, the absolute value of the classifier function $h(\mathbf{x})$ increases as one moves away from the classification hyperplane. The gradient $\nabla h(\mathbf{x})$ is hence positive, respectively negative, as one moves inside the region of the positive, respectively negative, class. We can exploit this observation to deflect selective components of the velocity signal from the original DS along, respectively opposite to, the direction $\nabla h(\mathbf{x})$. Concretely, if $\dot{\mathbf{x}}_O = f_{\text{sgn}(h(\mathbf{x}))}(\mathbf{x})$ denotes the velocity obtained from the original DS and

$$\lambda(\mathbf{x}) = \begin{cases} \max(\epsilon, \nabla h(\mathbf{x})^T \dot{\mathbf{x}}_O) & \text{if } h(\mathbf{x}) > 0 \\ \min(-\epsilon, \nabla h(\mathbf{x})^T \dot{\mathbf{x}}_O) & \text{if } h(\mathbf{x}) < 0 \end{cases}, \quad (3.1)$$

the modulated dynamical system is given by

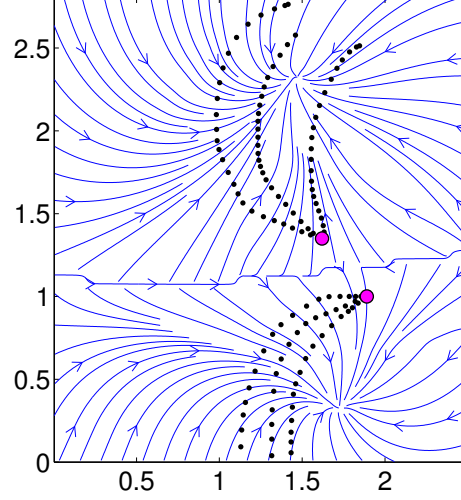
$$\dot{\mathbf{x}} = \tilde{f}(\mathbf{x}) = \lambda(\mathbf{x}) \nabla h(\mathbf{x}) + \dot{\mathbf{x}}_{\perp}. \quad (3.2)$$

Here, ϵ is a small positive scalar and $\dot{\mathbf{x}}_{\perp} = \dot{\mathbf{x}}_O - \left(\frac{\nabla h(\mathbf{x})^T \dot{\mathbf{x}}_O}{\|\nabla h(\mathbf{x})\|^2} \right) \nabla h(\mathbf{x})$ is the component of the original velocity perpendicular to ∇h . This results in a vector field that flows along increasing values of the classifier function in the regions of space where $h(\mathbf{x}) > 0$ and along decreasing values for $h(\mathbf{x}) < 0$. As a result, the trajectories move away from the classification hyperplane and converge to a point located in the region where they were initialized. Such modulated systems have been used extensively for estimating stability regions of interconnected power networks (Lee, 2003) and are known as *quasi gradient systems* (Chiang and Chu, 1996). If $h(\mathbf{x})$ is upper bounded¹, all trajectories converge to one of the stationary points $\{\mathbf{x} : \nabla h(\mathbf{x}) = 0\}$ and $h(\mathbf{x})$ is a Lyapunov function of the overall system (refer Chiang and Chu, 1996, proposition 1). Figure 3.3 shows the result of applying the above modulation to our pair of DS. As expected, it forces the trajectories to flow along the gradient of the function $h(\mathbf{x})$. Although this solves the problem of “crossing-over” the boundary, the trajectories obtained are deficient in two major ways. They depart heavily from the original dynamics and do not terminate at the desired attractors. This is due to the fact that the function $h(\mathbf{x})$ used to modulate the DS was designed solely for classification and contained no information about the dynamics of the two original DS. In other words, the vector field given by $\nabla h(\mathbf{x})$ was not aligned with the flow of the training trajectories and the stationary points of the modulation function did not coincide with the desired attractors.

In subsequent sections, we show how we can learn a new modulation function which takes into account the three issues we highlighted in this preliminary discussion. We will seek a system that *a)* ensures strict classification across regions

¹SVM classifier function is bounded if the Radial Basis Function (rbf) is used as kernel.

Figure 3.3 Trajectories obtained by modulating the two original DS with a SVM classifier function. The resulting trajectories flow along directions which register an increase in the value of the classifier function which in turn leads to the bifurcation at the SVM decision boundary.



of attraction (ROA) for each DS, *b*) follows closely the dynamics of each DS in each ROA and *c*) ensures that all trajectories in each ROA reach the desired attractor. Satisfying requirements *a*) and *b*) above is equivalent to performing classification and regression simultaneously. We take advantage of the fact that the optimization in support vector classification and support vector regression have the same form to phrase our problem in a single constrained optimization framework. In the next sections, we show that in addition to the usual SVM support vectors (SVs), the resulting modulation function is composed of an additional class of SVs. We analyze geometrically the effect of these new support vectors on the resulting dynamics. While this preliminary discussion considered solely binary classification, we will now extend the problem to multi-class classification.

3.4 Problem Formulation

The N -dimensional state space of the system represented by $\mathbf{x} \in \mathbb{R}^N$ is partitioned into M different classes, one for each of the M motions to be combined. We collect trajectories in the state space, yielding a set of P data points $\{\mathbf{x}_i; \dot{\mathbf{x}}_i; l_i\}_{i=1 \dots P}$ where $l_i \in \{1, 2, \dots, M\}$ refers to the class label of each point². To learn the set of modulation functions $\{h_m(\mathbf{x})\}_{m=1 \dots M}$, we proceed recursively. We learn each modulation function in a one-vs-all classifier scheme and then compute the final modulation function $\tilde{h}(\mathbf{x}) = \max_{m=1 \dots M} h_m(\mathbf{x})$. In the multi-class setting, the behavior of avoiding boundaries is obtained if the trajectories move along *increasing* values of the function $\tilde{h}(\mathbf{x})$. To this effect, the deflection term $\lambda(\mathbf{x})$ presented in the binary case (Equation 3.1) becomes $\lambda(\mathbf{x}) = \max \left(\epsilon, \nabla \tilde{h}(\mathbf{x})^T \dot{\mathbf{x}}_O \right); \forall \mathbf{x} \in \mathbb{R}^N$. Next, we describe the procedure for learning a single $h_m(\mathbf{x})$ function.

²Bold faced fonts represent vectors. \mathbf{x}_i denotes the i -th vector and x_i denotes the i -th element of vector \mathbf{x} .

We follow the classical SVM formulation and lift the data into a higher dimensional feature space through the mapping $\phi : \mathbb{R}^N \mapsto \mathbb{R}^F$ where F denotes the dimension of the feature space. We also assume that each function $h_m(\mathbf{x})$ is linear in feature space, i.e., $h_m(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b$ where $\mathbf{w} \in \mathbb{R}^F, b \in \mathbb{R}$. We label the current ($m - th$) motion class as positive and all others negative such that the set of labels for the current sub-problem is given by

$$y_i = \begin{cases} +1 & \text{if } l_i = m \\ -1 & \text{if } l_i \neq m \end{cases} ; \quad i = 1 \dots P.$$

Also, the set indexing the positive class is then defined as $\mathcal{I}_+ = \{i : i \in [1, P]; l_i = m\}$. With this, we formalize the three constraints explained in [Section 3.3](#) as:

Classification Each point must be classified correctly yields P constraints

$$y_i (\mathbf{w}^T \phi(\mathbf{x}_i) + b) \geq 1 \quad \forall i = 1 \dots P. \quad (3.3)$$

Lyapunov Constraint The gradient of the modulation function must have a positive component along the velocities at the data points. This ensures that the modulated flow is aligned with the training trajectories. We have the constraint

$$\nabla h_m(\mathbf{x}_i)^T \hat{\mathbf{x}}_i = \mathbf{w}^T \mathbf{J}(\mathbf{x}_i) \hat{\mathbf{x}}_i \geq 0 \quad \forall i \in \mathcal{I}_+ \quad (3.4)$$

where $\mathbf{J} \in \mathbb{R}^{F \times N}$ is the Jacobian matrix given by $\mathbf{J} = \left[\nabla \phi_1(\mathbf{x}) \nabla \phi_2(\mathbf{x}) \dots \nabla \phi_F(\mathbf{x}) \right]^T$ and $\hat{\mathbf{x}}_i = \dot{\mathbf{x}}_i / \|\dot{\mathbf{x}}_i\|$ is the normalized velocity at the $i - th$ data point.

Stability The gradient of the modulation function must vanish at the attractor \mathbf{x}^* of the positive class. This constraint can be expressed as

$$\nabla h_m(\mathbf{x}^*)^T \mathbf{e}_i = \mathbf{w}^T \mathbf{J}(\mathbf{x}^*) \mathbf{e}_i = 0 \quad \forall i = 1 \dots N \quad (3.5)$$

where the set of vectors $\{\mathbf{e}_i\}_{i=1 \dots N}$ is the canonical basis of \mathbb{R}^N .

3.4.1 PRIMAL & DUAL FORMS

As in the standard SVM ([Schölkopf and Smola, 2001](#)), we optimize for maximal margin between the positive and negative classes, subject to constraints [3.3-3.5](#) above. This can be formulated as:

$$\underset{\mathbf{w}, \xi_i}{\text{minimize}} \quad \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i \in \mathcal{I}_+} \xi_i$$

$$\text{subject to } \left. \begin{aligned} y_i (\mathbf{w}^T \phi(\mathbf{x}_i) + b) &\geq 1 & \forall i = 1 \dots P \\ \mathbf{w}^T \mathbf{J}(\mathbf{x}_i) \hat{\mathbf{x}}_i + \xi_i &> 0 & \forall i \in \mathcal{I}_+ \\ \xi_i &> 0 & \forall i \in \mathcal{I}_+ \\ \mathbf{w}^T \mathbf{J}(\mathbf{x}^*) \mathbf{e}_i &= 0 & \forall i = 1 \dots N \end{aligned} \right\}. \quad (3.6)$$

Here $\xi_i \in \mathbb{R}$ are slack variables that relax the Lyapunov constraint in Equation 3.4. We retain these in our formulation to accommodate noise in the data representing the dynamics. $C \in \mathbb{R}_+$ is a penalty parameter for the slack variables. The Lagrangian for the above problem can be written as

$$\begin{aligned} \mathcal{L}(\mathbf{w}, b, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) = & \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i \in \mathcal{I}_+} \xi_i - \sum_{i \in \mathcal{I}_+} \mu_i \xi_i - \sum_{i=1}^P \alpha_i (y_i (\mathbf{w}^T \phi(\mathbf{x}_i) + b) - 1) \\ & - \sum_{i \in \mathcal{I}_+} \beta_i (\mathbf{w}^T \mathbf{J}(\mathbf{x}_i) \hat{\mathbf{x}}_i + \xi_i) + \sum_{i=1}^N \gamma_i \mathbf{w}^T \mathbf{J}(\mathbf{x}^*) \mathbf{e}_i \end{aligned} \quad (3.7)$$

where $\alpha_i, \beta_i, \mu_i, \gamma_i$ are the Lagrange multipliers with $\alpha_i, \beta_i, \mu_i \in \mathbb{R}_+$ and $\gamma_i \in \mathbb{R}$. Employing a similar analysis as in the standard SVM, we derive the dual by setting the derivatives of the Lagrangian w.r.t all the variables and multipliers to zero, we get

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \mathbf{w}} &= \mathbf{w} - \sum_{i=1}^P \alpha_i y_i \phi(\mathbf{x}_i) - \sum_{i \in \mathcal{I}_+} \beta_i \mathbf{J}(\mathbf{x}_i) \hat{\mathbf{x}}_i + \sum_{i=1}^N \gamma_i \mathbf{J}(\mathbf{x}^*) \mathbf{e}_i = 0. \\ \Rightarrow \mathbf{w} &= \sum_{i=1}^P \alpha_i y_i \phi(\mathbf{x}_i) + \sum_{i \in \mathcal{I}_+} \beta_i \mathbf{J}(\mathbf{x}_i) \hat{\mathbf{x}}_i - \sum_{i=1}^N \gamma_i \mathbf{J}(\mathbf{x}^*) \mathbf{e}_i; \end{aligned} \quad (3.8)$$

$$\frac{\partial \mathcal{L}}{\partial b} = \sum_{i=1}^P \alpha_i y_i = 0; \quad (3.9)$$

$$\frac{\partial \mathcal{L}}{\partial \xi_i} = C - \beta_i - \mu_i = 0 \quad \forall i \in \mathcal{I}_+. \quad (3.10)$$

Combining 3.10 with the constraints that all the Lagrange multipliers β_i and μ_i be positive, we obtain

$$0 \leq \beta_i \leq C \quad \forall i \in \mathcal{I}_+. \quad (3.11)$$

Using 3.8, 3.9 and 3.10 in 3.7 we get the dual objective function to be maximized as

$$\hat{\mathcal{L}}(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) = \sum_{i=1}^P \alpha_i - \frac{1}{2} \mathbf{w}^T \mathbf{w}. \quad (3.12)$$

Note that although the dual has the same general form as the dual in the standard SVM formulation, it differs in the expression of the term \mathbf{w} . Expanding

using 3.8 we have

$$\begin{aligned}
\mathbf{w}^T \mathbf{w} &= \left(\sum_{i=1}^P \alpha_i y_i \phi(\mathbf{x}_i)^T + \sum_{i \in \mathcal{I}_+} \beta_i \hat{\mathbf{x}}_i^T \mathbf{J}(\mathbf{x}_i)^T - \sum_{i=1}^N \gamma_i \mathbf{e}_i^T \mathbf{J}(\mathbf{x}^*)^T \right) \\
&\quad \left(\sum_{j=1}^P \alpha_j y_j \phi(\mathbf{x}_j) + \sum_{j \in \mathcal{I}_+} \beta_j \mathbf{J}(\mathbf{x}_j) \hat{\mathbf{x}}_j - \sum_{j=1}^N \gamma_j \mathbf{J}(\mathbf{x}^*) \mathbf{e}_j \right) \\
&= \sum_{i=1}^P \left(\sum_{j=1}^P \alpha_i y_i \alpha_j y_j \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j) + \sum_{j \in \mathcal{I}_+} \alpha_i y_i \beta_j \phi(\mathbf{x}_i)^T \mathbf{J}(\mathbf{x}_j) \hat{\mathbf{x}}_j - \sum_{j=1}^N \alpha_i y_i \gamma_j \phi(\mathbf{x}_i)^T \mathbf{J}(\mathbf{x}^*) \mathbf{e}_j \right) \\
&\quad + \sum_{i \in \mathcal{I}_+} \left(\sum_{j=1}^P \beta_i \alpha_j y_j \hat{\mathbf{x}}_i^T \mathbf{J}(\mathbf{x}_i)^T \phi(\mathbf{x}_j) + \sum_{j \in \mathcal{I}_+} \beta_i \beta_j \hat{\mathbf{x}}_i^T \mathbf{J}(\mathbf{x}_i)^T \mathbf{J}(\mathbf{x}_j) \hat{\mathbf{x}}_j - \sum_{j=1}^N \beta_i \gamma_j \hat{\mathbf{x}}_i^T \mathbf{J}(\mathbf{x}_i)^T \mathbf{J}(\mathbf{x}^*) \mathbf{e}_j \right) \\
&\quad - \sum_{i=1}^N \left(\sum_{j=1}^P \gamma_i \alpha_j y_j \mathbf{e}_i^T \mathbf{J}(\mathbf{x}^*)^T \phi(\mathbf{x}_j) + \sum_{j \in \mathcal{I}_+} \gamma_i \beta_j \mathbf{e}_i^T \mathbf{J}(\mathbf{x}^*)^T \mathbf{J}(\mathbf{x}_j) \hat{\mathbf{x}}_j - \sum_{j=1}^N \gamma_i \gamma_j \mathbf{e}_i^T \mathbf{J}(\mathbf{x}^*)^T \mathbf{J}(\mathbf{x}^*) \mathbf{e}_j \right).
\end{aligned} \tag{3.13}$$

Rewriting in matrix form,

$$\mathbf{w}^T \mathbf{w} = [\boldsymbol{\alpha}^T \quad \boldsymbol{\beta}^T \quad \boldsymbol{\gamma}^T] \begin{bmatrix} \mathbf{K} & \mathbf{G} & -\mathbf{G}_* \\ \mathbf{G}^T & \mathbf{H} & -\mathbf{H}_* \\ -\mathbf{G}_*^T & -\mathbf{H}_*^T & \mathbf{H}_{**} \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\beta} \\ \boldsymbol{\gamma} \end{bmatrix} \tag{3.14}$$

where $\mathbf{K} \in \mathbb{R}^{P \times P}$, $\mathbf{G} \in \mathbb{R}^{P \times |\mathcal{I}_+|}$, $\mathbf{G}_* \in \mathbb{R}^{P \times N}$, $\mathbf{H} \in \mathbb{R}^{|\mathcal{I}_+| \times |\mathcal{I}_+|}$, $\mathbf{H}_* \in \mathbb{R}^{|\mathcal{I}_+| \times N}$, $\mathbf{H}_{**} \in \mathbb{R}^{N \times N}$ are given by

$$\left. \begin{aligned} [\mathbf{K}]_{ij} &= y_i y_j \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j) & [\mathbf{H}]_{ij} &= \hat{\mathbf{x}}_i^T \mathbf{J}(\mathbf{x}_i)^T \mathbf{J}(\mathbf{x}_j) \hat{\mathbf{x}}_j \\ [\mathbf{G}]_{ij} &= y_i \phi(\mathbf{x}_i)^T \mathbf{J}(\mathbf{x}_j) \hat{\mathbf{x}}_j & [\mathbf{H}_*]_{ij} &= \hat{\mathbf{x}}_i^T \mathbf{J}(\mathbf{x}_i)^T \mathbf{J}(\mathbf{x}^*) \mathbf{e}_j \\ [\mathbf{G}_*]_{ij} &= y_i \phi(\mathbf{x}_i)^T \mathbf{J}(\mathbf{x}^*) \mathbf{e}_j & [\mathbf{H}_{**}]_{ij} &= \mathbf{e}_i^T \mathbf{J}(\mathbf{x}^*)^T \mathbf{J}(\mathbf{x}^*) \mathbf{e}_j \end{aligned} \right\}. \tag{3.15}$$

where $[\cdot]_{ij}$ denotes the i, j -th entry of the corresponding matrix. Further using the relations A.2 and A.3, we can rewrite the above block matrices in terms of the kernel function and data:

$$\left. \begin{aligned} [\mathbf{K}]_{ij} &= y_i y_j k(\mathbf{x}_i, \mathbf{x}_j) & [\mathbf{H}]_{ij} &= \hat{\mathbf{x}}_i^T \frac{\partial^2 k(\mathbf{x}_i, \mathbf{x}_j)}{\partial \mathbf{x}_i \partial \mathbf{x}_j} \hat{\mathbf{x}}_j \\ [\mathbf{G}]_{ij} &= y_i \left(\frac{\partial k(\mathbf{x}_i, \mathbf{x}_j)}{\partial \mathbf{x}_j} \right)^T \hat{\mathbf{x}}_j & [\mathbf{H}_*]_{ij} &= \hat{\mathbf{x}}_i^T \frac{\partial^2 k(\mathbf{x}_i, \mathbf{x}^*)}{\partial \mathbf{x}_i \partial \mathbf{x}^*} \mathbf{e}_j \\ [\mathbf{G}_*]_{ij} &= y_i \left(\frac{\partial k(\mathbf{x}_i, \mathbf{x}^*)}{\partial \mathbf{x}^*} \right)^T \mathbf{e}_j & [\mathbf{H}_{**}]_{ij} &= \mathbf{e}_i^T \frac{\partial^2 k(\mathbf{x}^*, \mathbf{x}^*)}{\partial \mathbf{x}^* \partial \mathbf{x}^*} \mathbf{e}_j \end{aligned} \right\}. \tag{3.16}$$

These can be further expanded given a choice of the kernel. Expansions for the Radial Basis Function (rbf) and the non-homogeneous polynomial kernel are given in Appendix 2.

Using Equations 3.9, 3.11, 3.12 and 3.14 the dual optimization problem can be stated as

$$\underset{\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}}{\text{minimize}} \quad \frac{1}{2} [\boldsymbol{\alpha}^T \boldsymbol{\beta}^T \boldsymbol{\gamma}^T] \begin{bmatrix} \mathbf{K} & \mathbf{G} & -\mathbf{G}_* \\ \mathbf{G}^T & \mathbf{H} & -\mathbf{H}_* \\ -\mathbf{G}_*^T & -\mathbf{H}_*^T & \mathbf{H}_{**} \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\beta} \\ \boldsymbol{\gamma} \end{bmatrix} - \boldsymbol{\alpha}^T \bar{\mathbf{1}}$$

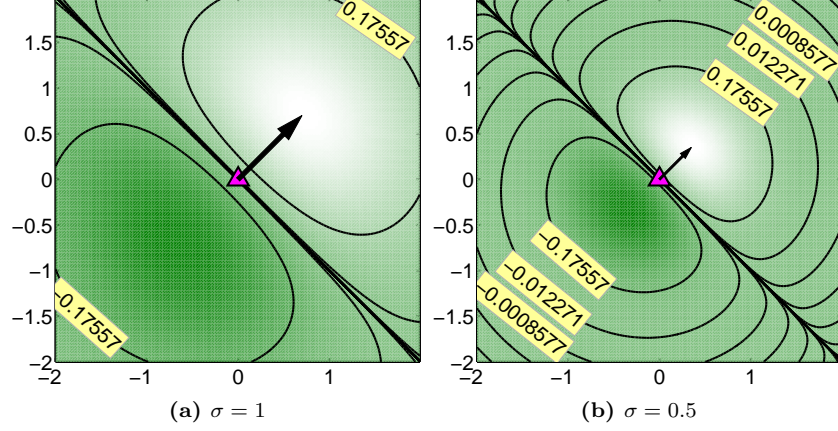


Figure 3.4: Isocurves of $f(\mathbf{x}) = \hat{\mathbf{x}}_i^T \frac{\partial k(\mathbf{x}, \mathbf{x}_i)}{\partial \mathbf{x}_i}$ at $\mathbf{x}_i = [0 \ 0]^T$, $\hat{\mathbf{x}}_i = [\frac{1}{\sqrt{2}} \ \frac{1}{\sqrt{2}}]^T$ for the rbf kernel with width σ .

subject to

$$\left. \begin{array}{ll} 0 \leq \alpha_i & \forall i = 1 \dots P \\ 0 \leq \beta_i \leq C & \forall i \in \mathcal{I}_+ \\ \sum_{i=1}^P \alpha_i y_i = 0 & \end{array} \right\}. \quad (3.17)$$

Note that the Lagrange multipliers γ_i are completely unconstrained as they correspond to the *equality* constraints in the primal. Also, since the matrices \mathbf{K} , \mathbf{H} and \mathbf{H}_{**} are symmetric, the overall Hessian matrix for the resulting quadratic program is also symmetric. In our implementation, we use the MATLAB[®] *quadprog* solver to solve this quadratic program. We initialize the iterations by setting α_i as the solution to the standard SVM classification problem. All β_i and γ_i are set to zeros. Once the optimal solution for the above problem is obtained, the modulation function can be written as

$$\begin{aligned} h(\mathbf{x}) &= \mathbf{w}^T \phi(\mathbf{x}) + b \\ &= \sum_{i=1}^P \alpha_i y_i \phi(\mathbf{x}_i)^T \phi(\mathbf{x}) + \sum_{i \in \mathcal{I}_+} \beta_i \hat{\mathbf{x}}_i^T \mathbf{J}(\mathbf{x}_i)^T \phi(\mathbf{x}) - \sum_{i=1}^N \gamma_i \mathbf{e}_i^T \mathbf{J}(\mathbf{x}^*)^T \phi(\mathbf{x}) + b \\ &= \sum_{i=1}^P \alpha_i y_i k(\mathbf{x}, \mathbf{x}_i) + \sum_{i \in \mathcal{I}_+} \beta_i \hat{\mathbf{x}}_i^T \frac{\partial k(\mathbf{x}, \mathbf{x}_i)}{\partial \mathbf{x}_i} - \sum_{i=1}^N \gamma_i \mathbf{e}_i^T \frac{\partial k(\mathbf{x}, \mathbf{x}^*)}{\partial \mathbf{x}^*} + b \end{aligned} \quad (3.18)$$

This modulation function has noticeable similarities with the standard SVM classifier function. The first summation term on the right hand side is composed of the α support vectors (α -SV) which act as support to the classification hyperplane. The second term entails a new class of support vectors that perform a linear combination of the normalized velocity $\hat{\mathbf{x}}_i$ at the training data points \mathbf{x}_i . These β support vectors (β -SVs) collectively contribute to the fulfilment of the Lyapunov constraint in Equation 3.4 by introducing a positive slope in the modulation function value along the directions $\hat{\mathbf{x}}_i$. Figure 3.4 shows the influ-

ence of a single β -SV for the rbf kernel $k(\mathbf{x}_i, \mathbf{x}_j) = e^{-1/2\sigma^2\|\mathbf{x}_i - \mathbf{x}_j\|^2}$ with \mathbf{x}_i at the origin and $\hat{\mathbf{x}}_i = [\frac{1}{\sqrt{2}} \frac{1}{\sqrt{2}}]^T$. Observe that the smaller the kernel width σ , the steeper the slope. The third summation term is a non-linear bias, which does not depend on the chosen support vectors, and performs a local modification around the desired attractor \mathbf{x}^* to ensure that the modulation function has a local maximum at that point. b is the constant bias which normalizes the classification margins as -1 and $+1$. We calculate its value by making use of the fact that for all the data points \mathbf{x}_i chosen as α -SV, we must have $y_i h_m(\mathbf{x}_i) = 1$. We use the average of the values obtained from different support vectors.

Figure 3.5 illustrates the effects of the support vectors in a 2D example by progressively adding them and overlaying the resulting DS flow in each case. The value of the modulation function $h_m(\mathbf{x})$ is shown by the color plot (white indicates high values). As the β -SVs are added - as shown in Figure 3.5(b) - they *force* the flow of trajectories along their associated directions. In Figs. 3.5(c)-(d), adding the two γ terms shifts the location of the maximum of the modulation function to coincide with the desired attractor. Once all the SVs have been taken into account, the streamlines of the resulting DS achieve the desired criteria, i.e., they follow the training trajectories and terminate at the desired attractor.

3.5 Application examples

In this section, we validate the presented A-SVM model on 2D (synthetic) data and on a robotic simulated experiment using a 7 degrees of freedom (DOF) KUKA-LWR arm mounted on a 3-DOF Omnirob base to catch falling objects. A video of the robotic experiment - simulated and real - is provided at the project url <http://asvm.epfl.ch>. Next, we present a cross-validation analysis of the error introduced by the modulation in the original dynamics. A sensitivity analysis of the region of attraction of the resulting dynamical system with respect to the model parameters is also presented. We used the rbf kernel for all the results presented in this section. As discussed in Section 3.3, the RBF kernel is advantageous as it ensures that the function $h_m(\mathbf{x})$ is bounded. To generate an initial estimate of each individual dynamical system, we used the technique proposed in Khansari Zadeh and Billard (2011a).

3.5.1 2D EXAMPLE

Figure 3.6 shows a synthetic example with 4 motion classes, each generated from a different closed form dynamics and containing 160 data points. The color plot indicates the value of the combined modulation function $\tilde{h}(\mathbf{x}) = \max_{m=1 \dots M} h_m(\mathbf{x})$ where each of the functions $h_m(\mathbf{x})$ are learned using the presented A-SVM technique. A total of 9 support vectors were obtained which is $< 10\%$ of the number of training data points. The trajectories obtained after modulating the original dynamical systems flow along increasing values of the modulation function,

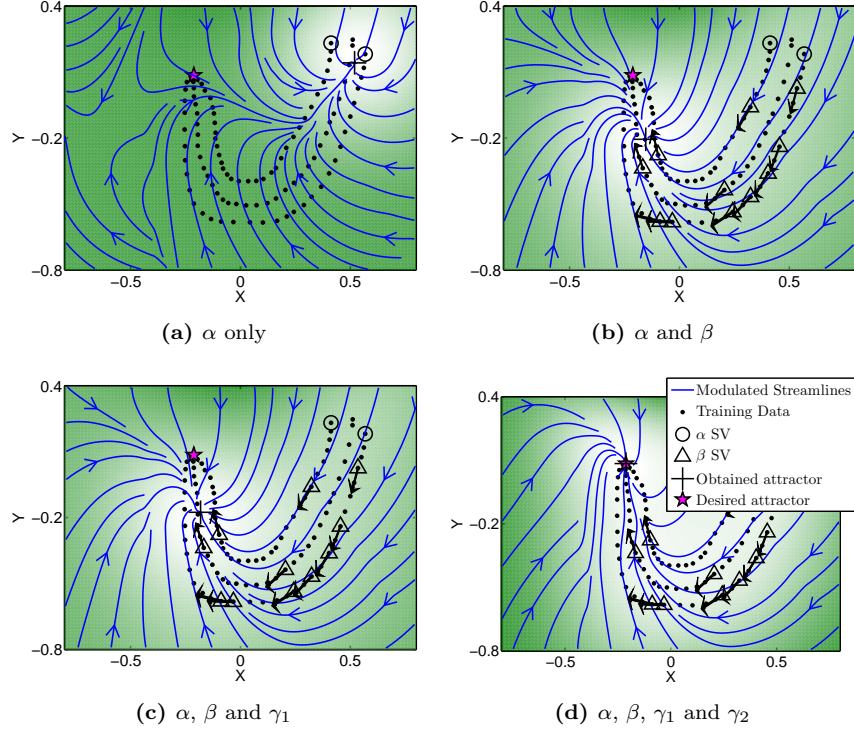


Figure 3.5: Progressively adding support vectors to highlight their effect on shaping the dynamics of the motion. (a) - α -SVs largely affect classification. (b) - β -SVs guide the flow of trajectories along their respective associated directions $\hat{\mathbf{x}}_i$ shown by arrows. (c)-(d) The 2 γ terms force the local maximum of the modulation function to coincide with the desired attractor along the X and Y axes respectively.

thereby bifurcating towards different attractors at the region boundaries. Unlike the dynamical system in Figure 3.3, the flow here is aligned with the training trajectories and terminates at the desired attractors. To recall, this is made possible thanks to the additional constraints (Equation 3.4 and 3.5) in our formulation.

In a second example, we tested the ability of our model to accommodate a higher density of attractors. We created 8 synthetic dynamics by capturing motion data using a screen mouse. Figure 3.1 shows the resulting 8 attractor system.

3.5.2 ERROR ANALYSIS

As formulated in Equation 3.6, the Lyapunov constraints admit some slack, which allows the modulation to introduce slight deviations from the original dynamics. Here we statistically analyze this error via 5-fold cross validation. In the 4 attractor problem presented above, we generate a total of 10 trajectories per motion class and use 2:3 training to testing ratio for cross validation.

We calculate the average percentage error between the original velocity (read off from the data) and the modulated velocity (calculated using Equation 3.2)

Figure 3.6 Resulting flow of the synthetic 4-attractor example. Color plot depicts the value of the 1-vs-all classifier function which has local maxima at all the 4 attractors.

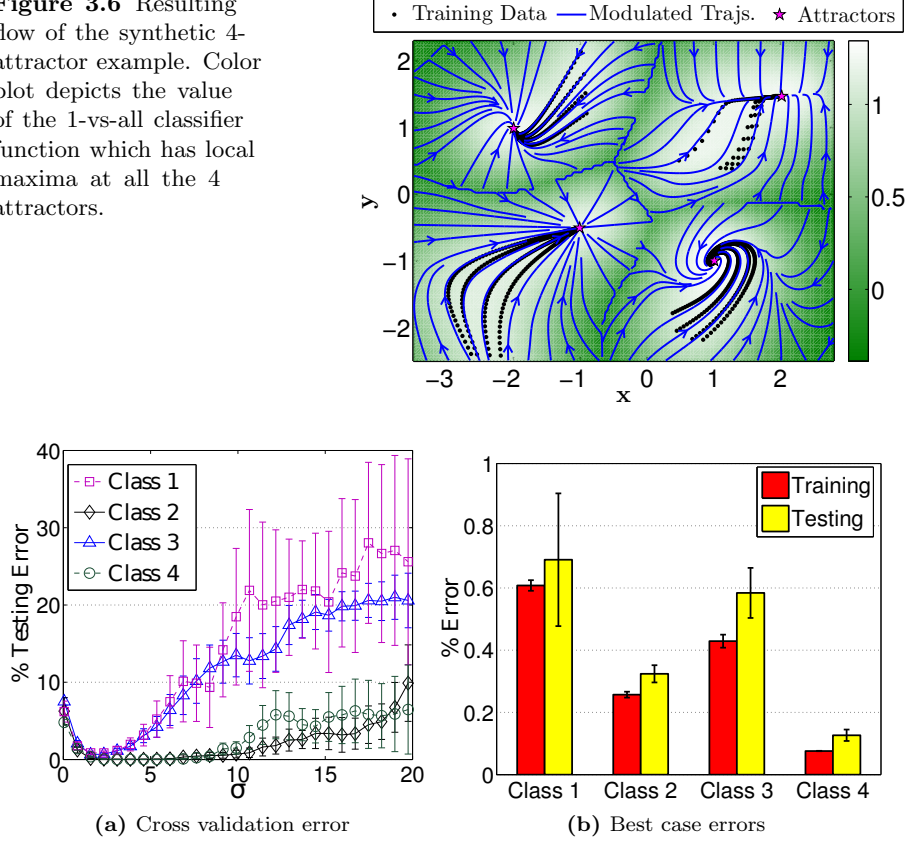


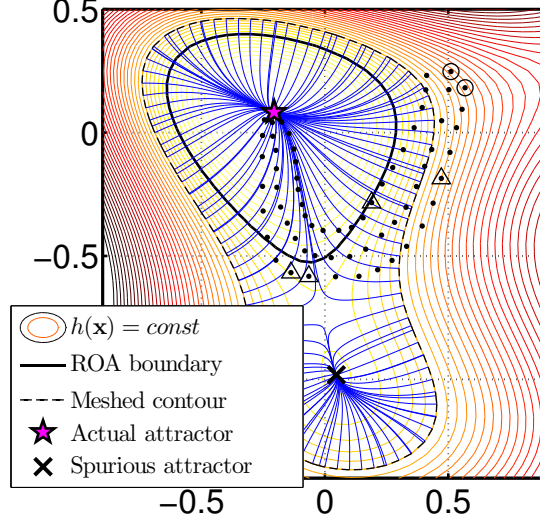
Figure 3.7: Error analysis for the synthetic 4-attractor example.

for the m -th class as $e_m = \left\langle \frac{\|\dot{\mathbf{x}}_i - \hat{f}(\mathbf{x}_i)\|}{\|\dot{\mathbf{x}}_i\|} \times 100 \right\rangle_{i:l_i=m}$ where $\langle . \rangle$ denotes average over the indicated range. Figure 3.7(a) shows the cross validation error (mean and standard deviation over the 5 folds) for a range of values of kernel width. The general trend revealed here is that for each class of motion, there exists a band of optimum values of the kernel width for which the testing error is the smallest. The region covered by this band of optimal values may vary depending on the relative location of the attractors and other data points. In Figure 3.6, motion classes 2 (upper left) and 4 (upper right) are better fitted and show less sensitivity to the choice of kernel width than classes 1 (lower left) and 3 (lower right). We will show later in this section that this is correlated to the distance between the attractors. A comparison of testing and training errors for the least error case is shown in Figure 3.7(b). We see that the testing errors for all the classes in the best case scenario are less than 1%.

3.5.3 SENSITIVITY ANALYSIS

The partitioning of space created by our method results in M regions of attraction (ROA) for each of our M attractors. To assess the size of these regions and the existence of spurious attractors, we adopt an empirical approach. For

Figure 3.8 Test trajectories starting from several points on an isocurve (dotted line) to determine spurious attractors.



each class, we compute the isosurfaces of the corresponding modulation function $h_m(\mathbf{x})$ in the range $[0, h_m(\mathbf{x}^*)]$. These hypersurfaces incrementally span the volume of the m -th region around its attractor. We mesh each of these test surfaces and compute trajectories starting from the obtained mesh-points, looking for spurious attractors. h_{ROA} is the isosurface of maximal value that encloses no spurious attractor and marks the ROA of the corresponding motion dynamics. We use the example in Figure 3.5 to illustrate this process. Figure 3.8 shows a case where one spurious attractor is detected using a larger test surface (dotted line) whereas the actual ROA (solid line) is smaller. Once h_{ROA} is calculated, we define the size of ROA as $r_{ROA} = (h(\mathbf{x}^*) - h_{ROA})/h(\mathbf{x}^*)$. $r_{ROA} = 0$ when no trajectory except those originating at the attractor itself, lead to the attractor. $r_{ROA} = 1$ when the ROA is bounded by the isosurface $h(\mathbf{x}) = 0$. The size of the r_{ROA} is affected by both the choice of kernel width and the distance across nearby attractors. This is illustrated in Figure 3.13 using data points from class 1 of Figure 3.6 and translating the attractors so that they are either very far apart (left, distance $d_{att} = 1.0$) or very close to one another (right, $d_{att} = 0.2$). As expected, r_{ROA} increases as we reach the optimal range of parameters. Furthermore, when the attractors are farther apart, high values of r_{ROA} are obtained for a larger range of values of the kernel width, i.e., the model is less sensitive to the chosen kernel width. With smaller distance between the attractors (Figure 3.13(b)), only a small deviation from the optimum kernel width results in a considerable loss in r_{ROA} , exhibiting high sensitivity to the model parameter.

3.5.4 3D EXAMPLE

We validated our method on a real world 3D problem. The attractors here represent manually labeled grasping points on a pitcher. The 3D model of the object was taken from the ROS IKEA object library. We use the 7-DOF

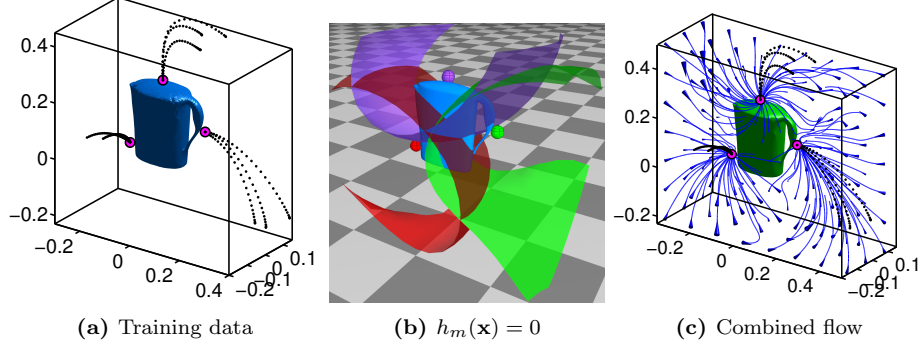


Figure 3.9: 3D Experiment. (a) shows training trajectories for three manually chosen grasping points. (b) shows the isosurfaces $h_m(\mathbf{x}) = 0; m = 1, 2, 3$ along with the locations of the corresponding attractors. (c) shows the complete flow of motion.

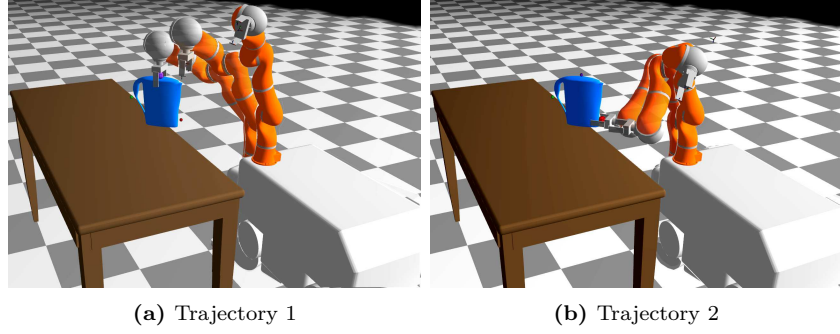


Figure 3.10: 10-DOF mobile robot executes the generated trajectories starting from different positions and hence converging to different grasping points (attractors).

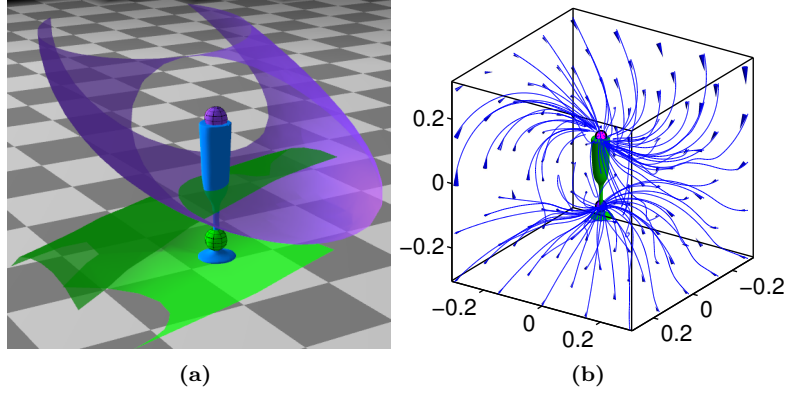
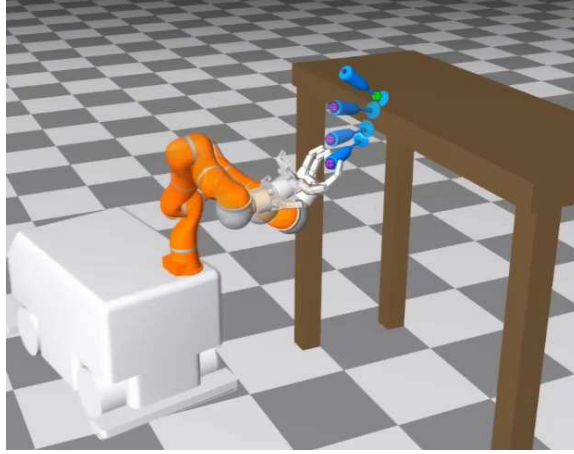
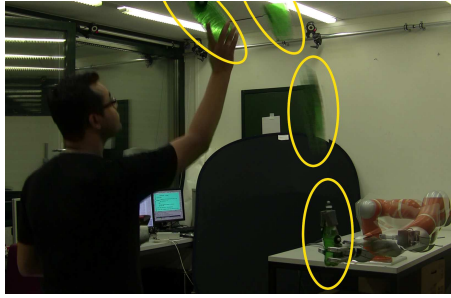


Figure 3.11: (a) Two attractors placed on a champagne glass and their corresponding classification surfaces. (b) Complete flow of motion around the object.

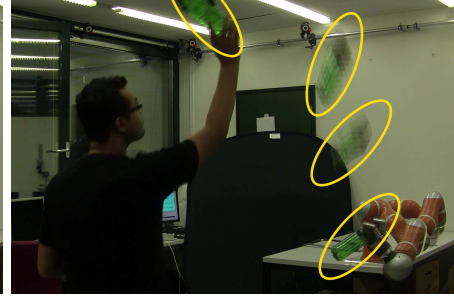
KUKA-LWR arm mounted on the 3-DOF KUKA-Omnibot base for executing the modulated Cartesian trajectories in simulation. We control all 10 DOF of the robot using the damped least square inverse kinematics. Training data for this implementation was obtained by recording the end-effector positions $\mathbf{x}_i \in \mathbb{R}^3$ from kinesthetic demonstrations of reach-to-grasp motions directed



(a)



(b)



(c)

Figure 3.12: (a) - Simulation experiment of catching a falling object with the 10-DOF KUKA-Omnibot. The robot switches between attractors (green to magenta) as the object falls down. (b), (c) - Real 7-DOF KUKA arm catching the falling object at different grasping points (attractors) in different throwing situations.

towards these grasping points, yielding a 3-class problem (see Figure 3.9(a)). Each class was represented by 75 data points. Figure 3.9(b) shows the isosurfaces $h_m(\mathbf{x}) = 0; m \in \{1, 2, 3\}$ learned using the presented method. Figure 3.10(a)-(b) show the robot executing two trajectories when started from two different locations and converging to a different attractor (grasping point). Figure 3.9(c) shows the flow of motion around the object. Note that the time required to generate each trajectory point is $O(S)$ where S denotes the total number of support vectors in the model. In this particular example with a total of 18 SVs, the trajectory points were generated at 1000 Hz which is well suited for real-time control. Such a fast generative model allows the robot to switch on-the-fly between the attractors and adapt to real-time perturbations in the object or the end-effector pose, without any re-planning or re-learning. Results for another object - a champagne glass with two attractors - are shown in Figure 3.11. We performed high speed experiments in which the glass is falling and the robot needs to catch it at one of the two attractors. This requires real-time adaptation to the constantly changing position and orientation of the object. The robot

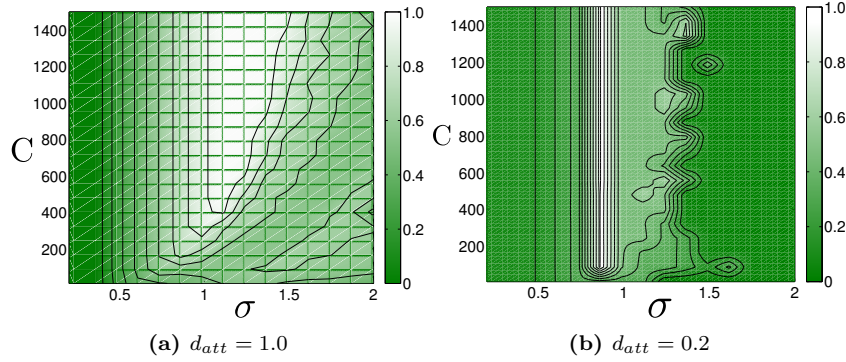


Figure 3.13: Variation of r_{ROA} with varying model parameters.

might need to switch between the attractors and move the end-effector toward the chosen attractor. Figure 3.12 shows the experiments in simulation and with the real KUKA robot. Full videos explaining the A-SVM methodology and these experiments are available at <http://asvm.epfl.ch/download.php>.

3.6 Conclusions

In this chapter we presented a SVM based learning framework for combining multiple non-linear dynamical systems through a partitioning of the space. The combined system is used to generate trajectories for reaching toward an object with multiple grasping points modelled as attractors, each with its own basin of attraction. Each of the basins are forward invariant with respect to the learned DS which ensures that the trajectories do not cross over the basin boundaries. We validated our method on synthetic 2D motions and 3D grasping motions on real objects. Results show that even though spurious attractors may occur, in practice they can be avoided by a careful choice of model parameters through grid search. The applicability of the method for real-time control of a 10-DOF robot was also demonstrated.

From a machine learning point of view, our framework is a reformulation of the SVM optimization framework which encapsulates gradient based constraints on the desired function. These constraints result in a new class of support vectors that exploit partial derivatives of the kernel function to align the flow of trajectories with the training data. In the next chapter we extend this framework to a general model-free function approximation technique which can handle simultaneous equality/inequality constraints on the function value and its gradient.

AUGMENTED-SVM FOR GRADIENT OBSERVATIONS IN FUNCTION APPROXIMATION

4.1 Foreword

The Augmented-SVM formulation presented in the previous chapter combined a set of constraints on the function value (arising from the classification requirement) with a set of constraints on the function gradient (arising from the energy requirement) into one optimization framework. In this chapter, we further generalize this framework for handling both equality and inequality constraints on the function value and its gradient. We analyze the form of the full optimization problem encompassing all possible constraints in terms of its convexity and give fast update rules with guaranteed convergence to solve it. Drawing parallels to the classical SVM formulations, we present the ν -parameterization which allows to have independent control over the complexity (number of support vectors) of the learned model. We show through realistic examples how this general formulation can be employed to any application where simultaneous constraints - equality and/or inequality - on the function value and its gradient are desired.

4.2 Introduction

Modelling the input-output response of a system from a few observations is commonly encountered in a variety of applications. In some of these applications, e.g., dynamic system identification, surface reconstruction, telemetry etc., observations of gradients of the output with respect to the input variables are available at each sampling of the system. In a typical regression setting, these observations are often not used for learning. Using this information to inform a learning algorithm would greatly reduce the number of samples required to achieve a desired level of accuracy. Although it comes at the cost of increasing the dimensionality of the learning problem, such an approach is highly desirable since the alternative is to sample more data which might be more expensive, and in some applications, practically infeasible. Furthermore, many dynamic systems exhibit known physical constraints in the form of local linearisations that must be respected by the approximation model. Again, obtaining a model that adheres to the physical constraints would require a denser sampling of the system if the constraints are not incorporated explicitly.

Figure 4.1 illustrates the improvement in model accuracy when using gradi-

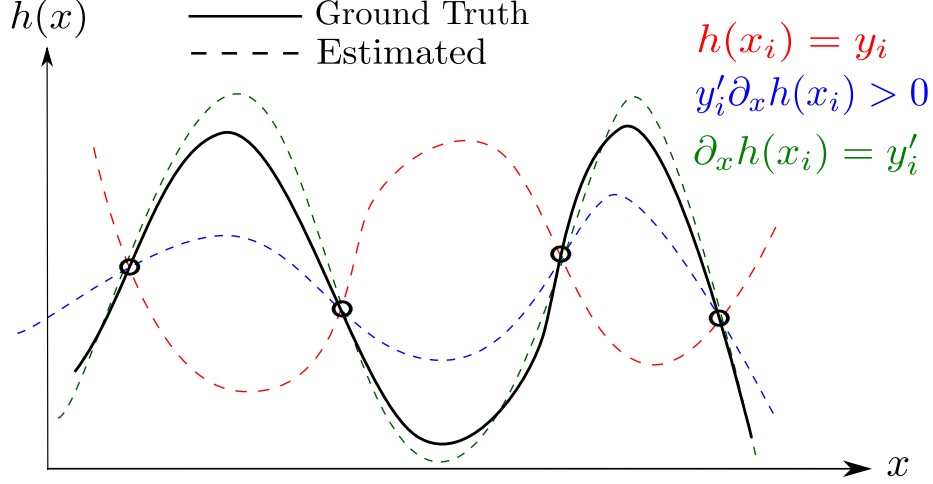


Figure 4.1: Adding gradient constraints in function approximation to improve accuracy with limited data.

ents for learning. When one uses only function values, one ends up with several equally good fits with respect to the training error (illustrated with the three dotted curves). Incorporating the additional knowledge of whether the function is increasing or decreasing at each input location rules out the estimate in red, since it does not conform with this new constraint. Such constraints may arise when we only have partial knowledge about the behaviour of a system and can only comment on the increasing/decreasing nature of the outputs instead of specifying the actual gradient. However, specifying the exact slope of the output function further removes the estimate in blue since it does not have the desired slopes at the input points.

Exploiting the availability of gradient information to learn more accurate representations with fewer training samples has been investigated in both application driven and theoretical contexts. The various possible formulations in this framework include classification type problems, which induce inequality constraints on the function value/gradient, or regression type problems, which induce equality constraints on the function value/gradient. Most previous works have considered a particular subset of such constraints depending on the target application. O’Hagan (1992) did pioneering work in using equality constraints on gradients in a Gaussian process (GP) setting which was followed by applications in control and engineering Murray-Smith et al. (1999). Equality constraints on function value and gradient were considered by Solak et al. (2003) in a GP framework. Although with the advantage of automatic determination of learning parameters, their formulations were restricted to equality constraints only. Micchelli and Pontil (2005); Macêdo and Castro (2008) used matrix-valued kernels for learning curl-free or divergence-free fluid flows. Their method focussed on estimating the gradient field of an underlying function using noisy observations of *only* the function gradient. Macêdo et al. (2009); Dragiev

et al. (2011) used hermite learning for interpolating contact points and tangents for learning implicit surfaces. They achieved high quality surface reconstruction but the results were restricted to interpolation in a noise-free setting. Rosasco et al. (2010) combined equality constraints on the function value with penalty on large values of the partial derivatives and applied it to variable selection. Their approach can be considered as one enforcing soft equality (to zero) constraints on the gradient. Shukla and Billard (2012b) combined inequality constraints on the function value with inequality and equality constraints on the gradient to represent a classifier/energy function for motion trajectories. Some recent works (Lemme et al., 2014; Reinhart and Steil, 2011) encode gradient constraints in a reservoir network for learning vector fields.

In the presented work, we unify all such formulations into one SVM framework for learning *any* combination of equality and/or inequality constraints on the function value and/or its directional derivative¹ Also, unlike many previous approaches that directly use the representer theorem, we use the KKT-Lagrangian machinery to derive our formulations. This further enables us to derive the fast SMO-like iterates for solving the resulting optimization problem and the ν parametrization as in the classical SVM.

This chapter is structured as follows. In Section 4.3, we first use a small set of constraints to introduce our working framework and the development of the primal-dual formulation. Subsequently, we give the complete formulation with all the constraints - inequality/equality constraints on the function value *combined* with inequality/equality constraints on the function gradient - in a single unified dual optimization problem. In Section 4.3.3 we show that the resulting dual optimization is convex and derive the globally convergent SMO-like Platt (1998) iterates that guarantee a decrease in the objective function at each iteration. In Section 4.3.4 we develop the ν parameterization in a way similar to the classical SVM Schölkopf et al. (2000) and establish that the parameter ν is a lower bound on the fraction of support vectors. Section 4.4 presents performance analyses highlighting the importance of gradient constraints in learning and the ability of the proposed formulation to incorporate gradient information to statistically decrease the testing errors.

4.3 Gradient formulations in SVM

For the sake of clarity, the formulations presented here will be without slack variables or noise assumptions. As in the classical SVM formulations, the presence of slack variables only translates into box constraints on the dual variables and noise assumptions add a complimentary set of Lagrange multipliers, e.g., α, α^* . While this increases the number of optimization variables, the derivation follows the same steps. The supplementary code we provide, however, has the

¹The directional derivative constraint subsumes the partial derivative case which has been considered in previous works.

full implementation.

4.3.1 PARTIAL FORMULATION

First, we consider an intuitive case with only equality constraints on the function value and its gradient. We wish to estimate a scalar function $h : \mathbb{R}^N \mapsto \mathbb{R}$ using the observed function values $y_i \in \mathbb{R}$ at input locations $\mathbf{z}_i \in \mathbb{R}^N$ and observed gradients $\mathbf{d}_i \in \mathbb{R}^N$ at input locations $\mathbf{x}_i \in \mathbb{R}^N$. For generality, we do not enforce that the function values and gradients be available at the same input points. The primal constraints for this problem are

$$\begin{aligned} h(\mathbf{z}_i) &= y_i \quad \forall i = 1 \cdots M_z \\ \nabla h(\mathbf{x}_i) &= \mathbf{d}_i \quad \forall i = 1 \cdots M_x. \end{aligned} \quad (4.1)$$

Following the SVM notation, we consider a projection of data in an F -dimensional feature space $\phi : \mathbb{R}^N \mapsto \mathbb{R}^F$. We assume that the function is linear in the feature space, i.e., $h(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b$ where $\mathbf{w} \in \mathbb{R}^F, b \in \mathbb{R}$. This further gives $\nabla h(\mathbf{x}) = \mathbf{J}(\mathbf{x})^T \mathbf{w}$ where $\mathbf{J} \in \mathbb{R}^{F \times N}$ is the Jacobian of the feature transformation. The primal problem is given by

$$\min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w}\|^2 \quad \text{subject to} \quad \begin{aligned} \mathbf{w}^T \phi(\mathbf{z}_i) + b &= y_i & i = 1 \cdots M_z \\ \mathbf{J}(\mathbf{x}_i)^T \mathbf{w} &= \mathbf{d}_i & i = 1 \cdots M_x. \end{aligned}$$

The corresponding Lagrangian can be written as

$$\mathcal{L}(\mathbf{w}, \alpha_i, \beta_i) = \frac{1}{2} \|\mathbf{w}\|^2 + \sum_{i=1}^{M_z} \alpha_i (y_i - \mathbf{w}^T \phi(\mathbf{z}_i) - b) + \sum_{i=1}^{M_x} (\mathbf{d}_i - \mathbf{J}(\mathbf{x}_i)^T \mathbf{w})^T \beta_i \quad (4.2)$$

where $\alpha_i \in \mathbb{R}$ and $\beta_i \in \mathbb{R}^N$ are the Lagrange multipliers. Note that we have a set of vector valued Lagrange multipliers β_i which correspond to the vector (gradient) constraints in the primal. Employing a similar analysis as in the standard SVM - writing the KKT conditions and rearranging terms - the corresponding dual is given by the constrained quadratic program:

$$\begin{aligned} \min_{\alpha, \beta} \frac{1}{2} [\alpha^T \beta^T] \begin{bmatrix} \mathbf{K} & \mathbf{G} \\ \mathbf{G}^T & \mathbf{H} \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} - \alpha^T \mathbf{y} - \sum_{i=1}^{M_x} \mathbf{d}_i^T \beta_i \\ \text{subject to } \alpha^T \mathbf{1} = 0. \end{aligned} \quad (4.3)$$

where $\beta \stackrel{\text{def}}{=} [\beta_1^T \cdots \beta_{M_x}^T]^T$. The block matrices $\mathbf{K} \in \mathbb{R}^{M_z \times M_z}, \mathbf{G} \in \mathbb{R}^{M_z \times NM_x}$ and $\mathbf{H} \in \mathbb{R}^{NM_x \times NM_x}$ are given by

$$[\mathbf{K}]_{ij} = \phi_i^T \phi_j ; [\mathbf{G}]_{ij} = \phi_i^T \mathbf{J}_j ; [\mathbf{H}]_{ij} = \mathbf{J}_i^T \mathbf{J}_j$$

where we have abbreviated $\phi(\mathbf{z}_i) \stackrel{\text{def}}{=} \phi_i$ and $\mathbf{J}(\mathbf{x}_i) \stackrel{\text{def}}{=} \mathbf{J}_i$. Using any valid Mercer kernel $k : \mathbb{R}^N \times \mathbb{R}^N \mapsto \mathbb{R}$ such that $k_{12} \stackrel{\text{def}}{=} k(\mathbf{x}_1, \mathbf{x}_2) = \phi^T(\mathbf{x}_1)\phi(\mathbf{x}_2)$ we can compute the above block matrices in terms of the data in the original space by using the relations

$$\phi_1^T \phi_2 = k_{12}; \mathbf{J}_1^T \phi_2 = \partial_{\mathbf{x}_1} k_{12}; \mathbf{J}_1^T \mathbf{J}_2 = \partial_{\mathbf{x}_1} \partial_{\mathbf{x}_2} k_{12}. \quad (4.4)$$

for any pair of vectors \mathbf{x}_1 and \mathbf{x}_2 .

After solving the dual for the optimal Lagrange multipliers, we recover the desired function as

$$h(\mathbf{x}) = \sum_{i=1}^{M_z} \alpha_i k(\mathbf{x}, \mathbf{x}_i) + \sum_{i=1}^{M_x} \beta_i^T \frac{\partial k(\mathbf{x}, \mathbf{x}_i)}{\partial \mathbf{x}_i} + b.$$

Note that the first summation term and the constant bias b in $h(\mathbf{x})$ above are the same as in the classical SVM classifier function. There are, however, new support terms accompanied by the Lagrange multipliers β_i . In the rest of the text, we will refer to this new set of support vectors as the β -SV. These Lagrange multipliers are vector valued and correspond to the gradient constraints in the primal. Furthermore, while the usual α -SV are characterized by the value of the kernel function centered at \mathbf{x}_i , the new β -SV are characterized by its first derivative.

4.3.2 COMPLETE FORMULATION

The above development gives a flavour of how the primal gradient constraints are processed through the Lagrangian and finally appear in the dual quadratic program, each with a corresponding vector valued Lagrange multiplier. Now we present the complete formulation and give only the main steps leading upto the final quadratic program. We use the following nomenclature: \mathbf{z}_i and \mathbf{x}_i respectively denote the points at which the function value and function gradient are constrained. Further, we use superscripts ‘=’ or ‘>’ to depict if the constraint is of type equality or in-equality respectively. Using this notation all the primal constraints can be divided into the following categories:

- Inequality constraints on the function value $\bar{y}_i h(\bar{\mathbf{z}}_i) \geq 1$ for $\bar{\mathbf{z}}_i \in \mathbb{R}^N$, $\bar{y}_i \in \mathbb{R}$; $i = [1, \bar{M}_z]$
- Equality constraints on the function value $h(\bar{\mathbf{z}}_i) = \bar{y}_i$ for $\bar{\mathbf{z}}_i \in \mathbb{R}^N$, $\bar{y}_i \in \mathbb{R}$; $i = [1, \bar{M}_z]$
- Inequality constraints on the function gradient $\bar{\mathbf{d}}_i^T \nabla h(\bar{\mathbf{x}}_i) \geq 0$ for $\bar{\mathbf{x}}_i \in \mathbb{R}^N$, $\bar{\mathbf{d}}_i \in \mathbb{R}^N$; $i = [1, \bar{M}_x]$
- Equality constraints on the function gradient $\nabla h(\bar{\mathbf{x}}_i) = \bar{\mathbf{d}}_i$ for $\bar{\mathbf{x}}_i \in \mathbb{R}^N$, $\bar{\mathbf{d}}_i \in \mathbb{R}^N$; $i = [1, \bar{M}_x]$

Combining these with regularization on the primal weights \mathbf{w} , we can write the Lagrangian as

$$\begin{aligned} \mathcal{L}(\cdot) = & \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^{\bar{M}_z} \bar{\alpha}_i \left[\bar{y}_i \left(\mathbf{w}^T \phi(\bar{\mathbf{z}}_i) + b \right) - 1 \right] + \sum_{i=1}^{\bar{M}_z} \bar{\alpha}_i \left[\bar{y}_i - \mathbf{w}^T \phi(\bar{\mathbf{z}}_i) - b \right] \\ & - \sum_{i=1}^{\bar{M}_x} \bar{\beta}_i \mathbf{w}^T \mathbf{J}(\bar{\mathbf{x}}_i) \bar{\mathbf{d}}_i + \sum_{i=1}^{\bar{M}_x} \left(\bar{\mathbf{d}}_i - \mathbf{J}(\bar{\mathbf{x}}_i)^T \mathbf{w} \right)^T \bar{\beta}_i \quad (4.5) \end{aligned}$$

using the Lagrange multipliers $\bar{\alpha}_i, \bar{\beta}_i \in \mathbb{R}_+$, $\bar{\alpha}_i \in \mathbb{R}$ and $\bar{\beta}_i \in \mathbb{R}^N$. Note that $\bar{\beta}_i$ is a vector Lagrange multiplier since it corresponds to a vector constraint. Setting derivatives to zero and re-arranging we get the dual program

$$\begin{aligned} \min_{\substack{\bar{\alpha}, \bar{\alpha} \\ \bar{\beta}, \bar{\beta}}} & \frac{1}{2} \begin{bmatrix} \bar{\alpha}^T & \bar{\alpha}^T & \bar{\beta}^T & \bar{\beta}^T \end{bmatrix} \mathbf{Q} \begin{bmatrix} \bar{\alpha} \\ \bar{\alpha} \\ \bar{\beta} \\ \bar{\beta} \end{bmatrix} - \bar{\alpha}^T \mathbf{1} - \bar{\alpha}^T \bar{\mathbf{y}} - \sum_{i=1}^{\bar{M}_x} \bar{\beta}_i^T \bar{\mathbf{d}}_i \\ \text{subject to} & \quad \bar{\alpha}_i \geq 0 ; \quad \bar{\beta}_i \geq 0 ; \quad \bar{\alpha}^T \bar{\mathbf{y}} + \bar{\alpha}^T \mathbf{1} = 0 \quad (4.6) \end{aligned}$$

where

$$\mathbf{Q} = \begin{bmatrix} \bar{\bar{K}} & \bar{\bar{K}} & \bar{\bar{G}} & \bar{\bar{G}} \\ & \bar{\bar{K}} & \bar{\bar{G}} & \bar{\bar{G}} \\ & & \bar{\bar{H}} & \bar{\bar{H}} \\ & & & \bar{\bar{H}} \end{bmatrix} \quad (4.7)$$

is a symmetric positive-semi-definite matrix whose internal blocks are given by

$$\begin{aligned} \bar{\bar{K}}_{ij} &= \bar{y}_i \bar{y}_j \phi(\bar{\mathbf{z}}_i)^T \phi(\bar{\mathbf{z}}_j) \quad ; \quad \bar{\bar{K}}_{ij} = \bar{y}_i \phi(\bar{\mathbf{z}}_i)^T \phi(\bar{\mathbf{z}}_j) \\ \bar{\bar{K}}_{ij} &= \phi(\bar{\mathbf{z}}_i)^T \phi(\bar{\mathbf{z}}_j) \quad ; \quad \bar{\bar{G}}_{ij} = \bar{y}_i \phi(\bar{\mathbf{z}}_i)^T \mathbf{J}(\bar{\mathbf{x}}_j) \bar{\mathbf{d}}_j \\ \bar{\bar{G}}_{ij} &= \bar{y}_i \phi(\bar{\mathbf{z}}_i)^T \mathbf{J}(\bar{\mathbf{x}}_j) \quad ; \quad \bar{\bar{G}} = \bar{\mathbf{d}}_j^T \mathbf{J}(\bar{\mathbf{x}}_j)^T \phi(\bar{\mathbf{z}}_i) \\ \bar{\bar{G}}_{ij} &= \phi(\bar{\mathbf{z}}_i)^T \mathbf{J}(\bar{\mathbf{x}}_j) \quad ; \quad \bar{\bar{H}}_{ij} = \bar{\mathbf{d}}_i^T \mathbf{J}(\bar{\mathbf{x}}_i)^T \mathbf{J}(\bar{\mathbf{x}}_j) \bar{\mathbf{d}}_j \\ \bar{\bar{H}}_{ij} &= \mathbf{J}(\bar{\mathbf{x}}_i)^T \mathbf{J}(\bar{\mathbf{x}}_j) \bar{\mathbf{d}}_j \quad ; \quad \bar{\bar{H}}_{ij} = \mathbf{J}(\bar{\mathbf{x}}_i)^T \mathbf{J}(\bar{\mathbf{x}}_j) \end{aligned}$$

which can be further written in terms of the chosen kernel and the data using [Equation 4.4](#). Solving the dual quadratic program in [4.6](#) for the optimal Lagrange multipliers, $h(\mathbf{x})$ can then be expressed as

$$h(\mathbf{x}) = \sum_{i=1}^{\bar{M}_z} \bar{\alpha}_i \bar{y}_i k(\mathbf{x}, \bar{\mathbf{z}}_i) + \sum_{i=1}^{\bar{M}_z} \bar{\alpha}_i k(\mathbf{x}, \bar{\mathbf{z}}_i) + \sum_{i=1}^{\bar{M}_x} \bar{\beta}_i^T \bar{\mathbf{d}}_i \frac{\partial k(\mathbf{x}, \bar{\mathbf{x}}_i)}{\partial \bar{\mathbf{x}}_i} + \sum_{i=1}^{\bar{M}_x} \bar{\beta}_i^T \frac{\partial k(\mathbf{x}, \bar{\mathbf{x}}_i)}{\partial \bar{\mathbf{x}}_i}. \quad (4.8)$$

4.3.3 SMO ITERATES

In this section we prove that the above dual optimization problem is convex and derive SMO-like iterates for solving it. We also show that these iterates result in non-increasing values of the objective function.

Proposition 1. *The Hessian matrix Q of the quadratic program in Equation 4.6 is positive semi-definite.*

Proof. We show that Q is a matrix of dot products (similar to the classical SVM case), and hence positive semi-definite. Defining $\mathbf{p}_i \stackrel{\text{def}}{=} \hat{y}_i \phi(\hat{\mathbf{z}}_i)$, $\mathbf{q}_i \stackrel{\text{def}}{=} \phi(\bar{\mathbf{z}}_i)$, $\mathbf{r}_i = J(\hat{\mathbf{x}}_i) \hat{\mathbf{d}}_i$, $\mathbf{s}_{i,j} = J(\bar{\mathbf{x}}_i) \mathbf{e}_j$, $\mathbf{s}_i \stackrel{\text{def}}{=} [\mathbf{s}_{i,1}^T \cdots \mathbf{s}_{i,N}^T]^T$ and substituting them in the expression for Q , we get

$$Q = \begin{bmatrix} [\mathbf{p}_i^T \mathbf{p}_j] & [\mathbf{p}_i^T \mathbf{q}_j] & [\mathbf{p}_i^T \mathbf{r}_j] & [\mathbf{p}_i^T \mathbf{s}_j] \\ & [\mathbf{q}_i^T \mathbf{q}_j] & [\mathbf{q}_i^T \mathbf{r}_j] & [\mathbf{q}_i^T \mathbf{s}_j] \\ & & [\mathbf{r}_i^T \mathbf{r}_j] & [\mathbf{r}_i^T \mathbf{s}_j] \\ & & & [\mathbf{s}_i^T \mathbf{s}_j] \end{bmatrix}.$$

Further defining

$$\mathbf{v} \stackrel{\text{def}}{=} \left[\mathbf{p}_1^T \cdots \mathbf{p}_{\hat{M}_z}^T, \mathbf{q}_1^T \cdots \mathbf{q}_{\bar{M}_z}^T, \mathbf{r}_1^T \cdots \mathbf{r}_{\hat{M}_x}^T, \mathbf{s}_1 \cdots \mathbf{s}_{\bar{M}_x} \right]^T$$

we get $Q = \mathbf{v} \mathbf{v}^T \succeq 0$. \square

For the full problem in 4.6, the number of optimization variables is $\hat{M}_z + N\bar{M}_z + \hat{M}_x + N\bar{M}_x$ which can be quite large for real-world data sets. Instead of applying classical gradient based optimization with line searches, SMO decomposes the full problem into a number of smallest possible quadratic problems, each of which have analytical solutions. It is known that the space and time complexity of SMO is linear in the number of training points (Platt, 1998) and hence it is highly useful to derive these updates for the new SVM formulations presented here. In the rest of this section, we derive the SMO updates for solving the quadratic program in 4.6.

Since our problem remains structurally similar to that of the classical SVM and $\hat{\alpha}_i$, $\bar{\alpha}_i$ are the Lagrange multipliers corresponding to the standard classification and regression constraints, the steps for deriving their updates remain exactly the same as in Platt (1998). Here we derive the updates for the remaining Lagrange multipliers.

$\hat{\beta}$ updates: The KKT optimality conditions for multipliers $\hat{\beta}_i$ are given by

$$\hat{\beta}_i = 0 \Leftrightarrow \nabla_{\hat{\mathbf{d}}_i}^T h(\hat{\mathbf{x}}_i) > 0 \text{ and } \hat{\beta}_i > 0 \Leftrightarrow \nabla_{\hat{\mathbf{d}}_i}^T h(\hat{\mathbf{x}}_i) = 0. \quad (4.9)$$

Since there are only box constraints on $\hat{\beta}_i$, the smallest solvable sub-problem is 1-dimensional. We iterate over all $\hat{\beta}_i$ and perform updates for only those which

are in violation of the above conditions. Assuming that the current Lagrange multiplier to be updated is $\overset{\rhd}{\beta}_l$, the objective function in Equation 4.6 can be written in terms of $\overset{\rhd}{\beta}_l$ as

$$\frac{1}{2} \overset{\rhd}{\overset{\rhd}{H}}_u (\overset{\rhd}{\beta}_l)^2 + B \overset{\rhd}{\beta}_l + \text{constant} \quad (4.10)$$

where $B \in \mathbb{R}$ is composed of all the coefficients of β_l in the expansion of the objective function in 4.6. Collecting those terms we get

$$\begin{aligned} B = & \sum_{i=1}^{\overset{\rhd}{M}_z} \overset{\rhd}{\alpha}_i \overset{\rhd}{y}_i \overset{\rhd}{\mathbf{d}}_l^T \frac{\partial k(\overset{\rhd}{\mathbf{x}}_l, \overset{\rhd}{\mathbf{z}}_i)}{\partial \overset{\rhd}{\mathbf{x}}_l} + \sum_{i=1}^{\bar{\overset{\rhd}{M}}_z} \bar{\overset{\rhd}{\alpha}}_i \bar{\overset{\rhd}{\mathbf{d}}}_l^T \frac{\partial k(\overset{\rhd}{\mathbf{x}}_l, \bar{\overset{\rhd}{\mathbf{z}}}_i)}{\partial \overset{\rhd}{\mathbf{x}}_l} \\ & + \sum_{\substack{i=1 \\ i \neq l}}^{\overset{\rhd}{M}_x} \overset{\rhd}{\beta}_i \overset{\rhd}{\mathbf{d}}_l^T \frac{\partial^2 k(\overset{\rhd}{\mathbf{x}}_l, \overset{\rhd}{\mathbf{x}}_i)}{\partial \overset{\rhd}{\mathbf{x}}_l \partial \overset{\rhd}{\mathbf{x}}_i} \overset{\rhd}{\mathbf{d}}_i + \sum_{i=1}^{\bar{\overset{\rhd}{M}}_x} \bar{\overset{\rhd}{\mathbf{d}}}_l^T \frac{\partial^2 k(\overset{\rhd}{\mathbf{x}}_l, \bar{\overset{\rhd}{\mathbf{x}}}_i)}{\partial \overset{\rhd}{\mathbf{x}}_l \partial \bar{\overset{\rhd}{\mathbf{x}}}_i} \bar{\overset{\rhd}{\beta}}_i \end{aligned} \quad (4.11)$$

Also, differentiating 4.8 with respect to $\overset{\rhd}{\mathbf{x}}_l$ evaluated at $\overset{\rhd}{\mathbf{x}}_l$ we get

$$\begin{aligned} \frac{\partial h(\overset{\rhd}{\mathbf{x}}_l)}{\partial \overset{\rhd}{\mathbf{x}}_l} = & \sum_{i=1}^{\overset{\rhd}{M}_z} \overset{\rhd}{\alpha}_i \overset{\rhd}{y}_i \frac{\partial k(\overset{\rhd}{\mathbf{x}}_l, \overset{\rhd}{\mathbf{z}}_i)}{\partial \overset{\rhd}{\mathbf{x}}_l} + \sum_{i=1}^{\bar{\overset{\rhd}{M}}_z} \bar{\overset{\rhd}{\alpha}}_i \frac{\partial k(\overset{\rhd}{\mathbf{x}}_l, \bar{\overset{\rhd}{\mathbf{z}}}_i)}{\partial \overset{\rhd}{\mathbf{x}}_l} \\ & + \sum_{i=1}^{\overset{\rhd}{M}_x} \overset{\rhd}{\beta}_i \frac{\partial^2 k(\overset{\rhd}{\mathbf{x}}_l, \overset{\rhd}{\mathbf{x}}_i)}{\partial \overset{\rhd}{\mathbf{x}}_l \partial \overset{\rhd}{\mathbf{x}}_i} \overset{\rhd}{\mathbf{d}}_i + \sum_{i=1}^{\bar{\overset{\rhd}{M}}_x} \frac{\partial^2 k(\overset{\rhd}{\mathbf{x}}_l, \bar{\overset{\rhd}{\mathbf{x}}}_i)}{\partial \overset{\rhd}{\mathbf{x}}_l \partial \bar{\overset{\rhd}{\mathbf{x}}}_i} \bar{\overset{\rhd}{\beta}}_i \end{aligned} \quad (4.12)$$

Combining 4.11 and 4.12 we get

$$B = \overset{\rhd}{\mathbf{d}}_l^T \left(\frac{\partial h(\overset{\rhd}{\mathbf{x}}_l)}{\partial \overset{\rhd}{\mathbf{x}}_l} - \overset{\rhd}{\beta}_l \frac{\partial^2 k(\overset{\rhd}{\mathbf{x}}_l, \overset{\rhd}{\mathbf{x}}_i)}{\partial \overset{\rhd}{\mathbf{x}}_l^2} \overset{\rhd}{\mathbf{d}}_l \right) = \overset{\rhd}{\mathbf{d}}_l^T \frac{\partial h(\overset{\rhd}{\mathbf{x}}_l)}{\partial \overset{\rhd}{\mathbf{x}}_l} - \overset{\rhd}{\beta}_l \overset{\rhd}{\overset{\rhd}{H}}_u.$$

Using the above relation with the closed-form minimizer of the quadratic Equation 4.11, we get the iterates as

$$\boxed{\overset{\rhd}{\beta}_l^{\text{new}} = -B / \overset{\rhd}{\overset{\rhd}{H}}_u = \overset{\rhd}{\beta}_l - \frac{\overset{\rhd}{\mathbf{d}}_l^T \nabla_{\overset{\rhd}{\mathbf{x}}_l} h(\overset{\rhd}{\mathbf{x}}_l)}{\overset{\rhd}{\overset{\rhd}{H}}_u}}. \quad (4.13)$$

In other words, the Lagrange multipliers $\overset{\rhd}{\beta}_l$ are updated proportionally to the normalized constraint violation. This means that the updates would increase the Lagrange multipliers for which the product $\overset{\rhd}{\mathbf{d}}_l^T \nabla_{\overset{\rhd}{\mathbf{x}}_l} h(\overset{\rhd}{\mathbf{x}}_l)$ is negative (i.e., points which violate the KKT constraint 4.9) and decreases those with positive dot products (i.e., those which do not violate 4.9) and potentially bring them down to 0, leading to a sparse solution. Lastly, it must be pointed out that the quadratic sub-problem in 4.10 is guaranteed to have a local minimum since the coefficient of the quadratic term $\overset{\rhd}{\overset{\rhd}{H}}_u = \overset{\rhd}{\mathbf{d}}_l^T \mathbf{J}(\overset{\rhd}{\mathbf{x}}_l) \mathbf{J}(\overset{\rhd}{\mathbf{x}}_l)^T \overset{\rhd}{\mathbf{d}}_l \geq 0 \quad \forall l$. This also

ensures that the value of the objective function does not increase on any update.

$\bar{\beta}$ updates: The KKT optimality condition for $\bar{\beta}_i$ is given by $\bar{\beta}_i = 0 \iff \nabla h(\bar{\mathbf{x}}_i) = \bar{\mathbf{d}}_i$. Note that here we have a vector valued Lagrange multiplier. Derivation for the SMO iterate in this case will be a vectorized version of the previous case. As above, since there are no primal constraints on $\bar{\beta}_i$, we can update each $\bar{\beta}_i$ separately. As before, we iterate over and update the Lagrange multipliers found in violation of the above condition. Let the currently chosen Lagrange multiplier to be updated is $\bar{\beta}_l$. Collecting the terms that contain $\bar{\beta}_l$, we can re-write the objective function as

$$\underbrace{\frac{1}{2} \bar{\beta}_l^T \bar{\mathbf{H}}_{ll} \bar{\beta}_l + B \bar{\beta}_l}_{\text{Quadratic term in Equation 4.6}} - \underbrace{\bar{\mathbf{d}}_l^T \bar{\beta}_l}_{\text{linear term in Equation 4.6}} + \text{constant} \quad (4.14)$$

where $B \in \mathbb{R}^N$ is given by

$$B = \sum_{i=1}^{\bar{M}_z} \bar{\alpha}_i \bar{y}_i \frac{\partial k(\bar{\mathbf{x}}_l, \bar{\mathbf{z}}_i)}{\partial \bar{\mathbf{x}}_l} + \sum_{i=1}^{\bar{M}_z} \bar{\alpha}_i \frac{\partial k(\bar{\mathbf{x}}_l, \bar{\mathbf{z}}_i)}{\partial \bar{\mathbf{x}}_l} + \sum_{i=1}^{\bar{M}_x} \bar{\beta}_i \frac{\partial k(\bar{\mathbf{x}}_l, \bar{\mathbf{x}}_i)}{\partial \bar{\mathbf{x}}_l \partial \bar{\mathbf{x}}_i} \bar{\mathbf{d}}_i + \sum_{i=1, i \neq l}^{\bar{M}_x} \frac{\partial k(\bar{\mathbf{x}}_l, \bar{\mathbf{x}}_i)}{\partial \bar{\mathbf{x}}_l \partial \bar{\mathbf{x}}_i} \bar{\beta}_i. \quad (4.15)$$

As before, differentiating the function $h(\mathbf{x})$ with respect to $\bar{\mathbf{x}}_l$ evaluated at $\bar{\mathbf{x}}_l$ we get

$$\frac{\partial h(\bar{\mathbf{x}}_l)}{\partial \bar{\mathbf{x}}_l} = \sum_{i=1}^{\bar{M}_z} \bar{\alpha}_i \bar{y}_i \frac{\partial k(\bar{\mathbf{x}}_l, \bar{\mathbf{z}}_i)}{\partial \bar{\mathbf{x}}_l} + \sum_{i=1}^{\bar{M}_z} \bar{\alpha}_i \frac{\partial k(\bar{\mathbf{x}}_l, \bar{\mathbf{z}}_i)}{\partial \bar{\mathbf{x}}_l} + \sum_{i=1}^{\bar{M}_x} \bar{\beta}_i \frac{\partial^2 k(\bar{\mathbf{x}}_l, \bar{\mathbf{x}}_i)}{\partial \bar{\mathbf{x}}_l \partial \bar{\mathbf{x}}_i} \bar{\mathbf{d}}_i + \sum_{i=1}^{\bar{M}_x} \frac{\partial^2 k(\bar{\mathbf{x}}_l, \bar{\mathbf{x}}_i)}{\partial \bar{\mathbf{x}}_l \partial \bar{\mathbf{x}}_i} \bar{\beta}_i. \quad (4.16)$$

Combining 4.15 and 4.16 gives

$$B = \frac{\partial h(\bar{\mathbf{x}}_l)}{\partial \bar{\mathbf{x}}_l} - \bar{\mathbf{H}}_{ll} \bar{\beta}_l. \quad (4.17)$$

The final update is given by combining the minimizer of Equation 4.14 given by $\bar{\beta}_l^{\text{new}} = -(\bar{\mathbf{H}}_{ll})^{-1} (B - \bar{\mathbf{d}}_l)$ with Equation 4.17 as

$$\boxed{\bar{\beta}_l^{\text{new}} = \bar{\beta}_l - (\bar{\mathbf{H}}_{ll})^{-1} \left(\frac{\partial h(\bar{\mathbf{x}}_l)}{\partial \bar{\mathbf{x}}_l} - \bar{\mathbf{d}}_l \right)}. \quad (4.18)$$

Again, the update is proportional and opposite in magnitude to the error in

the gradient at $\bar{\mathbf{x}}_l$, i.e, the data point corresponding to the current Lagrange multiplier. Also, the Hessian matrix of the quadratic equation in 4.14 is given by $\bar{\bar{H}}_{ll} = \mathbf{J}(\bar{\mathbf{x}}_l)^T \mathbf{J}(\bar{\mathbf{x}}_l)$ which is positive semi-definite, which in turn ensures that the objective function value is non-increasing on each update.

4.3.4 ν PARAMETERIZATION

Here we only give the analysis for the new Lagrange multipliers corresponding to the gradient constraints (equality and inequality). Analysis for the other two constraints remains the same as in classical SVM classification and regression cases. For deriving the ν -parameterized dual, as in the classical SVM, we must change the constraints to include noise parameters and slack variables. With overload of notation, let ' \succeq ' and ' \preceq ' respectively denote the element-wise ' \geq ' and ' \leq ' operators for vectors. The primal gradient constraints then become

$$\begin{aligned} -\bar{\xi}_i^* - \bar{\epsilon} \preceq \nabla h(\bar{\mathbf{x}}_i) - \bar{\mathbf{d}}_i \preceq \bar{\epsilon} + \bar{\xi}_i \quad & \left| \quad \begin{array}{l} \bar{\epsilon} \succeq \mathbf{0}, \bar{\epsilon} \geq 0 \\ \bar{\xi}_i^{(*)} \succeq \mathbf{0}, \bar{\xi}_i \geq 0 \end{array} \right. \\ \nabla h(\bar{\mathbf{x}}_i)^T \bar{\mathbf{d}}_i \geq \bar{\epsilon} - \bar{\xi}_i \end{aligned}$$

where $\bar{\epsilon} \in \mathbb{R}, \bar{\epsilon} \in \mathbb{R}^N$ denote the error-margins and $\bar{\xi}_i^{(*)} \in \mathbb{R}^N$ denotes the set of complementary slack variables $\bar{\xi}_i$ and $\bar{\xi}_i^*$. Similar to the classical ν -SVM, instead of using user-specified $\bar{\epsilon}$ and $\bar{\epsilon}$ we introduce new parameters² $\bar{\nu}_\beta \in \mathbb{R}^N, \bar{\nu}_\beta \in \mathbb{R}$ and minimize the quantity $\bar{\nu}_\beta^T \bar{\epsilon} - \bar{\nu}_\beta \bar{\epsilon}$ in the primal where $\mathbf{0} \preceq \bar{\nu}_\beta \preceq \mathbf{1}$ and $0 \leq \bar{\nu}_\beta \leq 1$. From the optimization point-of-view, in the inequality constraint, we are attempting to maximize the lower bound on the projection of the estimated function gradient onto the observed gradient. In the equality constraint, we are attempting to minimize the error margin (similar to the ϵ -tube width in SVR) in reconstruction of the gradient. Note that the new parameters $\bar{\nu}_\beta$ and $\bar{\nu}_\beta$ are more intuitive to specify since they are bounded. Moreover, as we will show later, they are also interpretable in terms of the minimum expected number of SV that arise due to the gradient constraints. The modified Lagrangian is then given by

$$\begin{aligned} \mathcal{L}(\cdot) = & \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^{\bar{M}_x} \bar{\beta}_i \left(\mathbf{w}^T \mathbf{J}(\bar{\mathbf{x}}_i) \bar{\mathbf{d}}_i - \bar{\epsilon} + \bar{\xi}_i \right) + \frac{1}{\bar{M}_x} \sum_{i=1}^{\bar{M}_x} \bar{\xi}_i - \bar{\mu} \bar{\epsilon} \\ & - \sum_{i=1}^{\bar{M}_x} \bar{\eta}_i \bar{\xi}_i - \bar{\nu}_\beta \bar{\epsilon} + \sum_{i=1}^{\bar{M}_x} \left(\mathbf{J}(\bar{\mathbf{x}}_i) \mathbf{w} - \bar{\mathbf{d}}_i \right)^T \left(\bar{\beta}_i - \bar{\beta}_i^* \right) + \bar{\nu}_\beta^T \bar{\epsilon} - \bar{\mu}^T \bar{\epsilon} \\ & \sum_{i=1}^{\bar{M}_x} \left(\frac{\bar{\xi}_i + \bar{\xi}_i^*}{\bar{M}_x} - \bar{\epsilon}^T (\bar{\beta}_i + \bar{\beta}_i^*) - \bar{\xi}_i^T \bar{\beta}_i - \bar{\xi}_i^{*T} \bar{\beta}_i^* - \bar{\eta}_i^T \bar{\xi}_i - \bar{\eta}_i^{*T} \bar{\xi}_i^* \right) \end{aligned}$$

²Similarly we can introduce $\bar{\nu}_\alpha, \bar{\nu}_\alpha \in \mathbb{R}$ which would be the same as the ' ν ' parameters of the standard SVM classification and regression respectively.

where the Lagrange multipliers $\bar{\eta}_i^{(*)}, \bar{\eta}_i, \bar{\mu}, \bar{\mu}$ correspond to the positivity constraints on $\bar{\xi}_i^{(*)}, \bar{\xi}_i, \bar{\epsilon}, \bar{\epsilon}$ respectively. Setting the derivatives of the Lagrangian with respect to $\bar{\epsilon}, \bar{\xi}_i^{(*)}$ and $\bar{\epsilon}, \bar{\xi}_i$ to zero we get respectively

$$\begin{aligned} \bar{\nu}_\beta - \sum_i \left(\bar{\beta}_i + \bar{\beta}_i^* \right) - \bar{\mu} &= \mathbf{0} & -\bar{\nu}_\beta + \sum_i \bar{\beta}_i - \bar{\mu} &= 0 \\ -\bar{\beta}_i^{(*)} + \frac{1}{\bar{M}_x} \mathbf{1} - \bar{\eta}_i^{(*)} &= \mathbf{0} & -\bar{\beta}_i + \frac{1}{\bar{M}_x} - \bar{\eta}_i &= 0 \end{aligned}$$

Setting the Lagrangian derivatives with respect to \mathbf{w} and b to zero gives the final objective function. Also, combining the above equations with positivity of the Lagrange multipliers $\bar{\eta}_i^* \succeq \mathbf{0}, \bar{\mu} \succeq \mathbf{0}$ and $\bar{\eta}_i \geq 0, \bar{\mu} \geq 0$ gives the final dual constraints. Skipping the details of these steps due to space constraints, the complete ν -parameterized dual can be written as

$$\begin{aligned} \min_{\substack{\bar{\beta}, \bar{\beta}^* \\ \bar{\beta} \succeq \bar{\beta}^*}} \quad & \frac{1}{2} \begin{bmatrix} \bar{\beta}^T & \bar{\beta}^T \end{bmatrix} \begin{bmatrix} \bar{H} & \bar{H} \\ \bar{H} & \bar{H} \end{bmatrix} \begin{bmatrix} \bar{\beta} \\ \bar{\beta} \end{bmatrix} + \sum_{i=1}^{\bar{M}_x} \left(\bar{\beta}_i + \bar{\beta}_i^* \right)^T \bar{\mathbf{d}}_i \\ & \text{subject to} \\ & \bar{\beta}_i^{(*)} \preceq \frac{1}{\bar{M}_x} \mathbf{1} ; \bar{\nu}_\beta \preceq \sum_i \left(\bar{\beta}_i + \bar{\beta}_i^* \right) ; \bar{\beta}_i \leq \frac{1}{\bar{M}_x} ; \bar{\nu}_\beta \leq \sum_i \bar{\beta}_i \end{aligned} \quad (4.19)$$

where $\bar{\beta} \stackrel{\text{def}}{=} \left[(\bar{\beta}_1 - \bar{\beta}_1^*)^T, \dots, (\bar{\beta}_{\bar{M}_x} - \bar{\beta}_{\bar{M}_x}^*)^T \right]^T$.

Proposition 2. $\bar{\nu}_\beta$ and $\bar{\nu}_\beta$ respectively lower bound the fraction of non-zero Lagrange multipliers $\bar{\beta}_i$ and $\bar{\beta}_i^*$ obtained in the optimal solution of the ν -parametrized dual in [Equation 4.19](#).

Proof. Let $\bar{\beta}_i(j)$ denote the j -th dimension of the Lagrange multiplier corresponding to the i -th gradient equality constraint. Since only one of $\bar{\beta}_i(j)$ or $\bar{\beta}_i^*(j)$ can be non-zero and from [4.19](#) we see that the maximal value of each $\bar{\beta}_i(j)$ or $\bar{\beta}_i^*(j)$ is $1/\bar{M}_x$, we get $\bar{\nu}_\beta(j) \preceq \sum_i (\bar{\beta}_i(j) + \bar{\beta}_i^*(j)) \preceq \frac{\bar{M}_\beta^j}{\bar{M}_x}$ where \bar{M}_β^j denotes the number of non-zero entries in the j -th dimension of the Lagrange multipliers $\bar{\beta}_i$. Summing over j both sides of the above inequality we get $\bar{\nu}_\beta^T \mathbf{1} \preceq \frac{\sum_j \bar{M}_\beta^j}{\bar{M}_x}$ and hence

$$\boxed{\bar{M}_\beta \geq \bar{M}_x \left(\bar{\nu}_\beta^T \mathbf{1} \right)}$$

where $\bar{M}_\beta = \sum_j \bar{M}_\beta^j$ denotes the total number of non-zero Lagrange multipliers corresponding to the gradient equality constraints.

A corollary to this is the fact that if all elements of the vector $\bar{\nu}_\beta$ are equal to some scalar $\bar{\nu}$, we get $\bar{M}_\beta \geq \bar{\nu} \left(N \bar{M}_x \right)$.

Similarly, for the second part, replacing each $\bar{\beta}_i$ with its maximal value

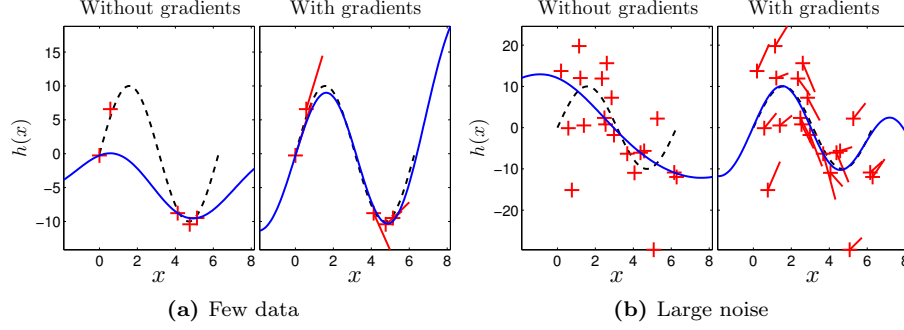


Figure 4.2: Learned estimates (solid blue line) of the underlying function (dotted black line) with and without gradient information. ‘+’ markers denote the noisy function value observations and the accompanying red lines denote the noisy slope information used for learning.

$1/\bar{M}_x$, we get $\bar{\nu}_\beta \leq \sum_i \bar{\beta}_i \leq \frac{\bar{M}_\beta}{\bar{M}_x}$ and hence

$$\boxed{\bar{M}_\beta \geq \bar{\nu}_\beta \bar{M}_x}$$

where \bar{M}_β denotes the total number of non-zero Lagrange multipliers corresponding to the gradient inequality constraints. \square

4.4 Applications

In this section, we instantiate the abstract formulations presented previously in the form of specific applications. We first apply the simple formulation presented in [Section 4.3](#) on a synthetic dataset to highlight the importance of gradient observations and validate our theoretical results. We also present two real-world examples - *a*) implicit surface reconstruction where the gradient observations appear as surface normals and *b*) estimating the energy function from a set of trajectories where the gradient observations appear as velocity vectors. All the results reported in the next sections can be reproduced from the MATLAB[®] scripts of the supplementary material.

4.4.1 SYNTHETIC EXAMPLE

We created a synthetic dataset by adding noise to a deterministic function

$$h(x) = 10\sin(x) + w_v \Rightarrow \nabla h(x) = 10\cos(x) + w_g \quad (4.20)$$

where $w_v \sim \mathcal{N}(0, \epsilon_v^2)$; $w_g \sim \mathcal{N}(0, \epsilon_g^2)$. ϵ_v and ϵ_g denote the noise variances in the function value and gradient observations respectively. First, we compare the learning output with and without gradient constraints in the two extreme

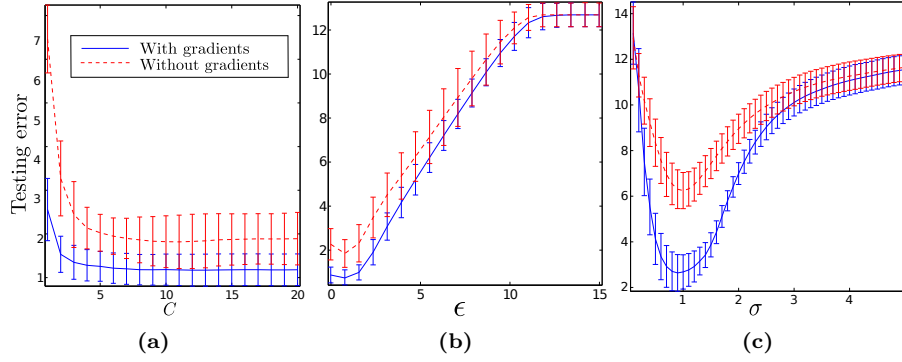


Figure 4.3: Variation of testing errors with training parameters.

(a)

	Few data	Large noise
w. grad.	0.72,0.73	0.54,0.64
w/o grad.	4.31,4.58	3.62,6.14

(b)

$\bar{\nu}_\alpha, \bar{\nu}_\beta$	0.1,0.1	0.1,0.9	0.9,0.1	0.9,0.9
$\bar{\nu}_\alpha \bar{M}_z, \bar{\nu}_\beta N \bar{M}_x$	2,2	2,18	18,2	18,18
$\bar{M}_\alpha, \bar{M}_\beta$	4,3	4,19	20,3	19,19

Table 4.1: (a) - Testing errors (value, gradient) corresponding to Figure 4.2. (b) - Variation of the number of support vectors with ν_α and ν_β for equality constraints.

cases - a) learning with extremely few data points and b) learning with sufficient data, but large noise. Figure 4.2(a) shows the case where we use only 5 data-points for training with $\epsilon_v = 1.0$ and $\epsilon_g = 1.0$ which is 10% of the function amplitude. Since there are so few data-points, their placement in the input space is extremely critical for learning the function sufficiently well. However, if gradient observations are available and incorporated in the model, sensitivity to placement of the data points is largely reduced. We used the radial basis function (rbf) kernel with kernel width $\sigma = 0.3$, penalty parameter $C = 100$ and the error-tube width for both function value and gradient $\epsilon = 1e - 3$ as training parameters. The corresponding testing errors are given in Table 4.1(a) which show a noticeable decrease in the testing error if gradients are used. We also see that the extrapolation (left and right extremes of the plot) is much more reliable if we take the gradient information into account. Figure 4.2(b) shows the learning outcome with sufficient data, but with much larger noise $\epsilon_v = 10.0$ and $\epsilon_g = 5.0$, i.e., 100% and 50% of the function amplitude. The corresponding testing errors are given in Table 4.1(a).

Next empirical results show that the learning output with gradients is less sensitive to the selection of training parameters. Figure 4.3 shows the variation of the testing error for the same dataset as above with the rbf kernel and different parameters C , ϵ and σ . In all the cases, we see that the testing error reduces by

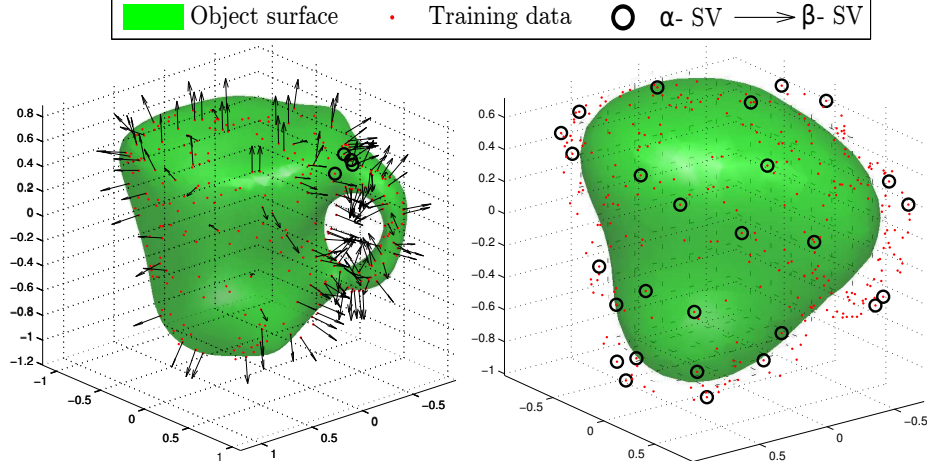


Figure 4.4: Surface reconstruction of a cup with (left) and without (right) gradient data. Not using gradient would require a larger data-set for getting the same quality of prediction.

a factor of $\approx 1/2$ if we use gradients for learning. As a consequence, the range of parameter values for acceptable performance is also increased. However, in Figure 4.3(a), we see that as ϵ increases this advantage is diminished and eventually both the testing errors converge to the same (large) testing error. This is expected since with an increase in ϵ , more and more data starts falling inside the error-margins and hence ignored. In Figure 4.3(c), similar effect is seen with increasing σ as well.

Lastly, in Table 4.1(b), we report empirical confirmation of the lower bound on the number of β -SV with varying $\bar{\nu}_\alpha$ and $\bar{\nu}_\beta$. Expectedly, as the parameters increase (row-1)³, the number of SVs (row-3) increase and are always greater than the predicted lower bound (row-2).

4.4.2 IMPLICIT SURFACE RECONSTRUCTION

In this experiment we apply the above formulation to learn a function that implicitly represents the surface of a real object as a particular iso-surface. We constrain the desired function to attain the value 1 at all the input points $\mathbf{x}_i \in \mathbb{R}^3$ and the gradient to be equal to the observed unit normals $\hat{\mathbf{n}}_i$ where $i = 1 \dots 300$. Translating this into our problem notation, we have $\bar{\mathbf{x}}_i = \bar{\mathbf{z}}_i = \mathbf{x}_i$, $\bar{y}_i = 1$ and $\bar{\mathbf{d}}_i = \hat{\mathbf{n}}_i$ which implies a total of 300 constraints on the function value (Lagrange multipliers $\bar{\alpha}_i$) and 300×3 constraints for the gradient (Lagrange multipliers $\bar{\beta}_i(j); j = 1, 2, 3$). In the optimal solution, we obtained only 4 and 374 SV for value and gradient constraints respectively. Figure 4.4 shows the final results obtained with and without gradient information. Notice the particularly high density of β -SV (arrows) near the handle due to the high non-linearity required in that region as against the cylindrical parts. The corresponding testing er-

³With slight abuse of notation, each element of the vector $\bar{\nu}_\beta$ is equal to the numbers in row-1 (0.1 or 0.9).

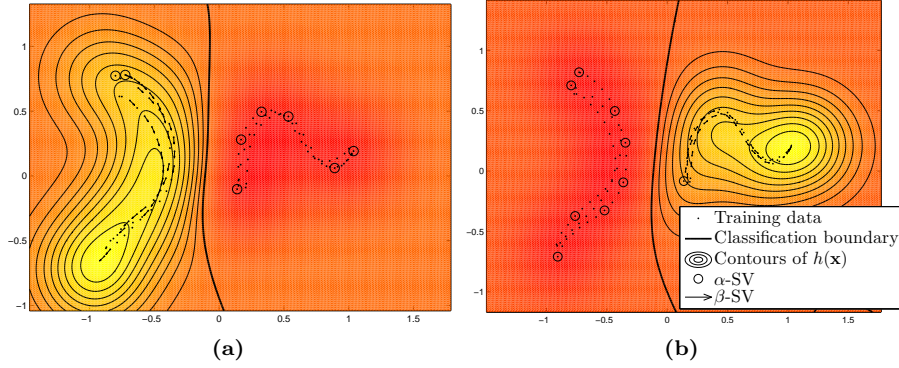


Figure 4.5: Learning a combined energy function and classifier for two sets of trajectories. The color plot depicts the value of the function which has stationary points (attractors) at the desired locations in each basin of attraction.

rors (value, gradient) are $(0.04, 0.25)$ and $(0.07, 1.57)$ for models learned with and without gradient respectively. Note that although the error on the function value is very similar in the two cases, the learning outcomes are however very different. This difference is explained when gradient errors are taken into account. This shows that the gradient errors are extremely important in quantifying how well the underlying function has been learned. We must also mention here that many surface exploration techniques, e.g., tactile exploration, provide information about surface normals together with the contact points at *each* exploration. If used, this information can greatly reduce the amount of exploration required.

4.4.3 ENERGY FUNCTION LEARNING

Here we give another instantiation of our formulation on the application of learning energy functions from trajectories of an underlying dynamical system (DS). We show that the problem and formulation presented in the previous chapter is a particular case of the general framework presented here. This problem has been visited by many researchers (Neumann et al., 2013; Shukla and Billard, 2012b; Prokhorov and Feldkamp, 1999; Serpen, 2005) using different function approximation techniques.

To recall from the previous chapter, energy functions grow monotonically along a trajectory $\{\mathbf{x}_t, \dot{\mathbf{x}}_t\}_{t=1\dots T}; \mathbf{x}_t \in \mathbb{R}^N$. This requires the function to obey a set of inequality constraints to ensure a positive projection of the gradient of the energy function onto the flow of the trajectory at each training point, i.e., $\nabla h(\mathbf{x}_t)^T \dot{\mathbf{x}}_t > 0$. Furthermore, gradient of the energy function must also vanish at the attractor $\mathbf{x}_* \in \mathbb{R}^N$ of the DS where all trajectories converge and terminate. This entails an equality constraint on the gradient, i.e., $\nabla h(\mathbf{x}_*) = \mathbf{0}$. Moreover, if the underlying DS has multiple attractors, a classification (inequality) constraint needs to be added to learn the separating boundary between the different basins of attraction. I.e., if $y_t \in \{+1, -1\}$ denote the basin of attraction in which the point \mathbf{x}_t lies, we must have $y_t h(\mathbf{x}_t) > 1$. Translating these

constraints to our formulation, we have $\vec{\mathbf{z}}_t, \vec{\mathbf{x}}_t \equiv \mathbf{x}_t; \vec{\mathbf{d}}_t \equiv \dot{\mathbf{x}}_t; \bar{\mathbf{x}}_1 \equiv \mathbf{x}^*; \bar{\mathbf{d}}_1 \equiv \mathbf{0}$. The resulting energy functions for such a multi-attractor system are shown in Figure 4.5. The function $h(\mathbf{x})$ that separates the two basins of attraction with the classification boundary also acts as the energy function that increases along the trajectories and has a stationary point at the desired attractors. In this case, the β -SV (shown by arrows) can be specifically interpreted as directions chosen by the optimization in which to bend the contours of the classifier function, as specified by the velocity vectors $\dot{\mathbf{x}}_t$.

4.5 Conclusions

In this chapter we generalized the Augmented-SVM framework of the previous chapter as a general function approximation technique. We showed that this framework can incorporate gradient information either from noisy observations or known physical constraints to enhance accuracy in model learning. Synthetic and real-world implementations show that including gradient information statistically improves the testing error and gives reliable prediction even with large noise and on very small-sized training sets. Although this advantage comes with an increase in computational cost by MN optimization variables - where M is the number of gradient constraints and N is the dimensionality of the input space - it is highly desirable in applications where gradient observations are available at each sampling of the system and the sampling process itself is expensive.

MOTION PLANNING ON ENERGY-MAPS LEARNED FROM HUMAN DEMONSTRATIONS

5.1 Foreword

The previous chapters advocated for and presented reactive, online, closed-loop flow based planning methods for reach-to-grasp motions. We showed that these methods offer a distinct advantage in terms of on-the-fly re-planning capability and implicitly encoding task constraints in a learned model. However, they trade-off the property of *probabilistic-completeness* for on-line adaptation, i.e., they are not guaranteed to find a feasible plan even if one exists. On the other end of the spectrum are sampling based motion planning algorithms which have been extremely successful in computing a feasible path in a non-convex space, such as those found in highly cluttered real-world environments. Most sampling based planners are global in nature and have the attractive property of being probabilistically complete. By the term “global” planner, we refer to planners that consider the whole workspace for computing a path toward the goal. This is in contrast with the local planners presented earlier in this thesis which only consider local information in the form of demonstrations. It is also worth mentioning here that the term “global” is not to be taken with respect to the optimality properties of the planner. Although these planners are relatively fast and capable of handling arbitrary environments, they fall under the planning-execution paradigm which is not ideal for dynamically changing environments. In this chapter, we present a combination of the two extremes which attempts to retain the good properties of both approaches, i.e., retain the probabilistic guarantees of global methods while significantly reducing the computational burden by utilizing the local information of the reactive methods.

5.2 Introduction

Owing to the conflicting but desirable properties of the global and local approaches to motion planning, a body of works aims at combining the best of both. The *randomly exploring random trees* (RRT) algorithm by [Kuffner and LaValle \(2000\)](#) is one of the well known probabilistically complete sampling based planners. Although originally designed with the goal of efficiently “searching” a space for a target configuration, many variations have been de-

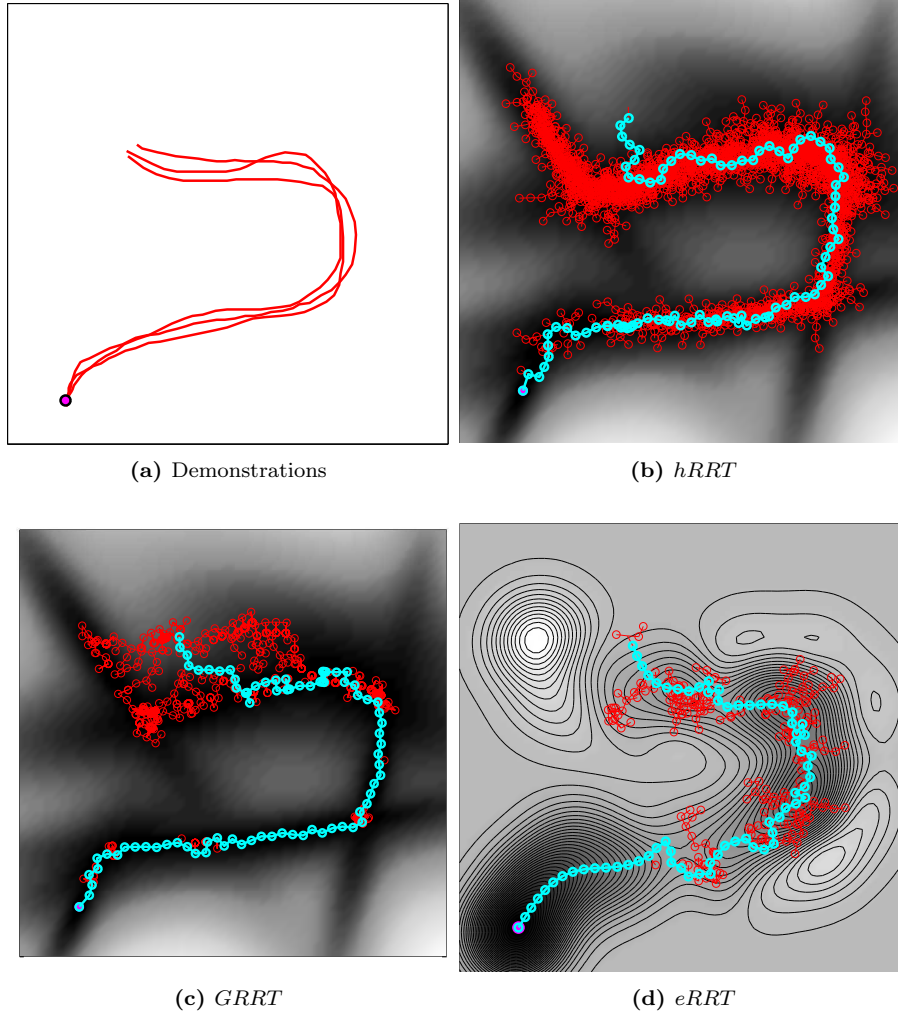


Figure 5.1: Different RRT variants searching on a 2-D cost map learned from the demonstrations. Darker regions represent smaller cost. (b) - $hRRT$ is extremely aggressive in terms of the cost and hence slow. (c) - $GRRT$ trades off cost aggressiveness with time efficiency using the manually set temperature scheduling parameters. (d) - The presented approach uses an energy map (instead of cost map) and exploration is bootstrapped with failures of the gradient step. Fast gradient steps of $eRRT$ expands nodes only where needed and reach the goal faster than other algorithms.

veloped to compute paths that are better suited for particular tasks. Most commonly used framework to incorporate task knowledge into planning is to assign a task-specific cost-map to the state space. The planner is designed to avoid large cost regions and is biased to grow the search tree toward low cost regions of the state space. The objective is to hence to search for paths with low cumulative cost. This chapter presents an alternative methodology for finding low cost paths based on energy functions rather than cost functions. We focus on the following two problems: a) how to model and learn an energy-function from demonstrations that is representative of a task and b) how to utilize the

additional information from the demonstrations to guide the search and decrease planning time.

Traditionally, cost functions were specified either in closed form by using geometric properties of the environment or, more recently, learned from a set of demonstrations. Learning the cost function from demonstrations provides a way to implicitly specify the task-constraints without specifically formulating them in terms of the state variables (Calinon and Billard, 2008). More recently, a number of inverse-reinforcement learning algorithms (Abbeel and Ng, 2004) are developed in which cost functions are represented as a weighted combination of features extracted from the demonstrations and could be learned by optimizing the weights. Once the cost function is known, the sampling based planner may be restricted to search in low cost areas. Not surprisingly, this additional constraint results in higher computational complexity, i.e., we trade off planning time with the quality of the path obtained. In this work, we investigate the effect of replacing the cost function with an energy function, which gives additional information to the planner regarding better directions to search from any given location. We show that such energy maps can be learned from demonstrations, and on contrary to cost-maps, planning on energy-maps results in a decrease in planning time.

As demonstrated in earlier chapters, learning such energy-maps is more complex than learning static cost-maps due to the presence of the additional energy-constraint, i.e., the energy function must decrease along the demonstrated trajectories. Representing this requirement mathematically requires constraints on the gradient of the energy function. In this chapter, we exploit and extend the formulation presented in Chapter 3 - which had a discrete set of point target states - to accommodate a continuous set of target states. From a planning perspective, this can be seen as a generalization of the concept of *workspace goal regions* presented in Berenson et al. (2009); Gienger et al. (2006) which are explicitly defined by the user in terms of the state-space variables. However, in our work, the target region is implicitly learned from demonstrations.

Due to the additional information encoded in the gradient, replacing cost-functions by energy-functions gives a distinct advantage in terms of estimating the *good* directions in which to expand the search tree. Gradient of the function can be used to guide the search and hence reducing the dependency on random explorations to find a path. This is our second contribution. Unlike typical RRT implementations which select nodes to be expanded using Voronoi bias, we select low-cost nodes with a higher priority and expand them in the direction suggested by the energy function. The priority of a node decreases if it fails to expand (due to collisions) and the search moves to a more exploratory mode. Exploration might lead to new low-cost nodes being created which expand successfully in the direction suggested by the model, in turn, creating nodes with even lower costs. In summary, our approach avoids unnecessary exploration steps. Instead, it invokes exploration only when it is needed and in the region where it is needed.

Moreover, unlike most previous RRT variants which control exploration using a hand-tuned temperature schedule, we bootstrap this parameter using a heuristic depending on the dimensionality of the state space. We use this heuristic to set the parameter in all the comparative experiments presented in this chapter.

This chapter is structured as follows. In the next section we review the related work in cost-map guided motion planning. [Section 5.4](#) presents the framework for learning energy-maps from demonstrations. [Section 5.5](#) presents formally our sampling based planning algorithm on energy-maps. We extend the algorithm to hybrid planning in [Section 5.6](#). Comparative results on toy problems and implementation on the real KUKA LWR platform is presented in [Section 5.7](#). Some drawbacks and directions for future improvements are discussed in [Section 5.8](#).

5.3 Literature Review

One of the first approaches introducing task-dependant bias in RRT exploration was the *heuristic-RRT* (*hRRT*) algorithm by [Urmson and Simmons \(2003\)](#). This approach was based on finding k-nearest neighbours of a random state at each expansion step and selecting the best quality node for expansion. This method produced exceptionally dense trees in low cost regions, which on one hand gives high quality (low cost) paths, and on the other hand results in an increase in planning time. A temperature dependent annealing-like exploration algorithm, *Transition-RRT* (*TRRT*) was presented by [Jaillet et al. \(2010\)](#). They interleaved basic RRT exploration with node rejection for expansions which resulted in an increase in the cost function. The probability of rejection was decreased as more nodes failed to expand. The temperature schedule and threshold for node-expansion failures were open parameters which traded off cost-aggressiveness of the search with computation time. This approach can be seen as a generalization of the previous *hRRT*, where the aggressiveness of the algorithm to look for low cost paths could be tuned using the temperature scheduling parameter. Other variants of the original *TRRT* were presented in [Devaurs et al. \(2013\)](#) for trade off between path quality and planning time. The performance of the different algorithms were found to be task dependant without any algorithm clearly outperforming others. In general, the *TRRT* algorithms were faster than previous alternatives. However, they suffered from difficulty in navigating through narrow cost spaces. This problem was addressed in [Berenson et al. \(2011\)](#) with the *Gradient-RRT* (*GRRT*) algorithm which was substantially faster than *TRRT* in narrow cost chasms. The algorithm proceeds with *TRRT* expansion steps, and on failure, expands the search tree deterministically in the direction of the negative gradient of the cost function. The key assumption here was that the negative gradient direction makes progress “towards” the goal and hence will help the nodes grow through narrow cost valleys. However, they used a Gaussian Mixture Model (GMM) to learn the cost function which

did not take this constraint into account. As shown in our results, many *GRRT* explorations “overlap” onto the tree since the negative gradient of the cost function is pointing in arbitrary directions. It is precisely this constraint, that we will model explicitly in our SVM based learning framework.

Few of the existing works in cost-based planning consider the problem of learning the cost itself. [Ettlin and Bleuler \(2006a,b\)](#) specify a closed form function for the cost depending on the “obstacleness” of a location on a rough terrain. They used geometric features of the terrain and combined to give a function that has large value on “difficult” regions of the state space. [Mainprice et al. \(2011\)](#) also specified a closed form cost function based on rest postures and potential energy of the links. [Berenson et al. \(2011\)](#) keep all the demonstrations to compute the a normalized distance function to be used as the cost-map. [Ye and Alterovitz \(2011\)](#) keep a part of the demonstrations as a guiding path and search for the complete path if the guiding path is split into multiple components due to collisions. They attempt to connect the disconnected components and construct a graph by sampling new nodes from the cost-map. A common feature among most of the above approaches is that they either specify heuristic functions as cost, or learn a cost function which is only used as a goodness measure of newly sample nodes. However, some works [Bowen et al. \(2014\)](#) have come up with cost functions which have a probabilistic basis and guarantee optimal paths. In our work, on the other hand, we learn a energy-function from demonstrations which in addition to being a goodness measure, also encodes an estimate of potentially better directions to search.

Another related body of planning with task information uses the Probabilistic Roadmaps ([Kavraki et al., 1996b](#)) (PRM). [Burns and Brock \(2005a\)](#) used a Gaussian Process to model the free space and add new nodes in the roadmap with probability inversely proportional to the variance of the model. Information theoretic models for guiding the search were presented in [Burns and Brock \(2005c,b\)](#); [Knepper and Mason \(2012\)](#). The sampler was biased toward new nodes that resulted in minimum entropy of the roadmap. In essence, these approaches do not use demonstrations, and hence can be useful in cases where little is known about the search space itself or providing demonstrations is not possible.

5.4 Energy function learning

We formally define an energy function in the context of a given set of trajectories as follows.

Definition 1. *Let a set of points $\mathbf{x}_i \in \mathbb{R}^N$ denote a trajectory sampled at specific time instants $i \in [0, T]$. A scalar valued function $h : \mathbb{R}^N \mapsto \mathbb{R}$ defines an energy map corresponding to this trajectory if the function decreases along the trajectory, i.e.,*

$$h(\mathbf{x}_{i+1}) < h(\mathbf{x}_i) \quad \forall t \in [0, T).$$

Corollary: If $\dot{\mathbf{x}}_i$ denotes the velocity at location \mathbf{x}_i , a valid energy function $h(\mathbf{x})$ will satisfy

$$\langle \nabla h(\mathbf{x}_i), \dot{\mathbf{x}}_i \rangle \leq 0 \quad \forall i \in [0, T)$$

where $\langle \cdot, \cdot \rangle$ denotes the internal product between two vectors.

For learning the energy function, we collect a set of demonstrations $\{\mathbf{x}_i, \dot{\mathbf{x}}_i\}_{i=1 \dots T}$ sampled at regular intervals along a demonstrated trajectory. For the toy examples presented in this chapter, we collected the demonstrations by recording a moving mouse pointer. For real/simulated robot tasks the demonstrations were collected via kinesthetic demonstrations. If $h(\mathbf{x}) \in \mathbb{R}$ denotes the desired energy function, \mathbf{x}^* denotes the desired target configuration and $\hat{\mathbf{x}}_i$ denotes the normalized velocity at each data point \mathbf{x}_i , we are looking for the following properties:

Region Separation: The function should have a lower value near the demonstrations as compared to other regions of the state space, i.e., $h(\mathbf{x}_i) < -1$ for each datapoint and $h(\mathbf{x}) = 0$ away from the demonstrations.

Energy constraint: The function should decrease in the direction of the demonstrated trajectories, i.e., $\langle \nabla h(\mathbf{x}_i), \hat{\mathbf{x}}_i \rangle < 0$.

Stability: In applications where a target configuration \mathbf{x}^* is specified, the function should have a local minimum, i.e., $\nabla h(\mathbf{x}^*) = \mathbf{0}$.

Note that the region separation constraint above is slightly different as compared to the that of [Chapter 3](#). In [Chapter 3](#) the constraint was one of binary classification, whereas here it is analogous to *one-class* classification where we separate the region of space containing demonstrations from the rest of the space. Next, we re-iterate the constraints and use the SVM framework of [Chapter 4](#) to formulate the problem.

Following the SVM methodology, we lift the data into a higher dimensional feature space through the mapping $\phi : \mathbb{R}^N \mapsto \mathbb{R}^F$ where F denotes the dimensionality of the feature space. Then, $h(\mathbf{x}) = \langle \mathbf{w}, \phi(\mathbf{x}) \rangle + b$ is the required energy function where $\mathbf{w} \in \mathbb{R}^F, b \in \mathbb{R}$. Moreover, $\nabla h(\mathbf{x}) = \mathbf{J}(\mathbf{x})^T \mathbf{w}$ where $\mathbf{J} \in \mathbb{R}^{F \times N}$ is the Jacobian of the feature space given by $[\phi_1(\mathbf{x}) \dots \phi_F(\mathbf{x})]^T$. Using these relations we can specify the primal optimization problem simply as $\min \frac{1}{2} \|\mathbf{w}\|^2$ under constraints

$$\begin{aligned} \langle \mathbf{w}, \phi(\mathbf{x}_i) \rangle &\leq -1 & i &= 1 \dots M \\ \mathbf{w}^T \mathbf{J}(\mathbf{x}_i) \hat{\mathbf{x}}_i &< 0 & i &= 1 \dots M \\ \mathbf{J}^T(\mathbf{x}^*) \mathbf{w} &= \mathbf{0}. \end{aligned} \tag{5.1}$$

Note that we have set $b = 0$ to satisfy the region separation constraint, i.e., $h(\mathbf{x}) = 0$ away from the demonstrations¹. The kernelized dual corresponding to

¹Many kernels such as the radial basis functions (rbf) and thin-plate splines have the property that they evaluate to zero far from the data. This ensures that the function $h(\mathbf{x})$ evaluates to b in regions where there are no demonstrations.

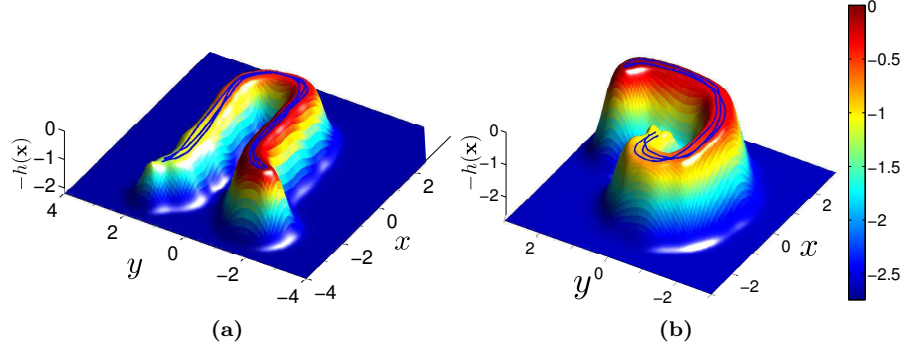


Figure 5.2: Learned energy functions from demonstrations (blue lines) for two different toy examples.

this primal is the following convex quadratic program:

$$\min_{\alpha, \beta, \gamma} \frac{1}{2} [\alpha^T \ \beta^T \ \gamma^T] Q \begin{bmatrix} \alpha \\ \beta \\ \gamma \end{bmatrix} - \alpha^T \mathbf{1} \text{ subject to } \begin{matrix} 0 < \alpha_i \\ 0 < \beta_i < C \end{matrix} \quad (5.2)$$

where α, β, γ are the vector of Lagrange multipliers corresponding to constraints 5.1 and the kernel matrix Q can be computed in terms of the kernel function and its derivatives. Development of the dual and expressions for computing the kernel matrix are detailed in Chapters 3 and 4. Optimal Lagrange multiplier values are obtained from the solution of the quadratic program in 5.2. The final energy function can then be written as

$$h(\mathbf{x}) = - \sum_i \left(\alpha_i k(\mathbf{x}, \mathbf{x}_i) + \beta_i \hat{\mathbf{x}}_i^T k'(\mathbf{x}, \mathbf{x}_i) \right) - \gamma^T k'(\mathbf{x}, \mathbf{x}^*) \quad (5.3)$$

where k' denotes the derivative of the kernel function with respect to the fixed variable (\mathbf{x}_i or \mathbf{x}^*). Figure 5.2 shows two examples of energy functions learned from demonstration trajectories in 2-dimensions². The demonstrations are shown in blue and overlaid on the obtained energy function. Negative of the function is plotted for visual clarity. It is evident from the surface plot that if one moves along the gradient, a trajectory starting away from the demonstrations moves toward the demonstration envelop and thereafter moves along the demonstrated path.

Another advantage of encoding the demonstrations as energy-map is that multiple goals can be specified in a single map. Such a scenario appears when there are many acceptable goal states and the planner may choose any one of them. In previous works, goal regions were explicitly represented by specifying bounds on the state space variables. Such an approach is only feasible when the goal regions to be specified are axis-aligned and a simple bound over variables suffices. Here we implicitly represent the goal regions in the form of multiple or

²All experiments in this chapter use the rbf kernel.

continuously connected local minima. [Figure 5.3](#) shows a the energy function for such a toy example. The demonstrations are shown as black lines which terminate on different locations. This may represent a continuous set of goal states forming a line. In [Section 5.7](#) we present a real-world example where two continuous patches serve as goal regions since they represent the graspable parts of an object.

Note that the learning procedure presented here is prone to local minima in the energy function. In the absence of local minima and obstacles, simply following the negative gradient of the energy function suffices to plan a path. However, in the presence of local minima or obstacles, we must interleave deterministic gradient following behaviour with the RRT exploration to compute a feasible path. Next, we describe an algorithm for planning feasible paths over the learned energy-maps.

5.5 Energy-RRT

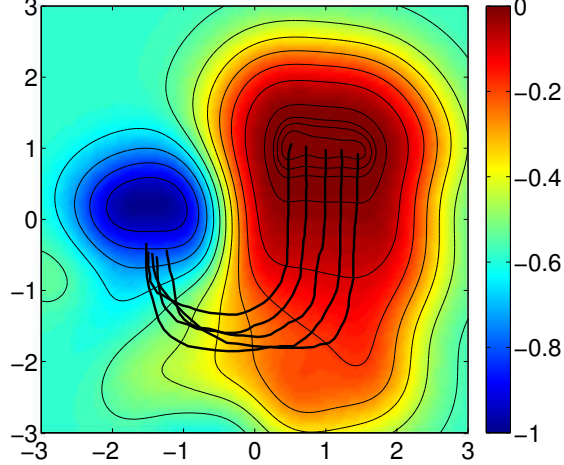
The algorithm presented here is tailored to exploit the properties of the learned energy-map. This comes from the fundamental assumption that the demonstrations were feasible trajectories, and as a result, the negative gradient of the energy function is oriented toward search directions which are most likely to succeed. The algorithm keeps a queue of the nodes sorted according to their priorities to keep track of the best (least energy) nodes. The priority α_i of a newly added node at location \mathbf{x}_i is set equal to the negative of the energy value at that node, i.e.

$$\alpha_i = -h(\mathbf{x}_i).$$

At each iteration, the algorithm tries to extend the highest priority node in the direction of the negative gradient. If this extension fails - due to a collision or a local minimum³ in the energy map - one RRT-like exploration step is made using a biased random sampler. The sampling is implemented by accepting the best (in terms of energy) out of N exploration steps. In effect, such a biased sampler weighs the Voronoi regions in favour of the low-cost regions of the state space [Urmson and Simmons \(2003\)](#). Moreover, at each failure, the priority of the failed node is decreased by a factor which allows subsequent exploration steps to move to regions of larger energy. Note that this step in the algorithm is analogous to the temperature schedule in other planners which controls the amount of failures which are *tolerated* before exploring high cost regions of space. As opposed to having an open parameter, we bootstrap the temperature schedule by using the dimensionality of the search space. The idea behind such an approach is that in high dimensions more explorations are needed before one can discard a node and move to higher cost nodes for exploration. The priority of a failed node in

³Local minimums are detected when taking a step along the negative gradient step results in a node with higher energy.

Figure 5.3 Toy example for a continuous goal region inferred from demonstrations. The colormap represents the negative energy function. There is a continuous set of local maxima representing the goal region.



the priority queue is decreased by a multiplicative factor as

$$\alpha_i^{new} = \alpha_i^{old} \sqrt[N]{\frac{h(\mathbf{x}_{start})}{h(\mathbf{x}_i)}}$$

where \mathbf{x}_{start} denotes the starting configuration from where a path is sought. This update rule is derived from the observation that after N updates, the priority of this node will be equal to the priority corresponding to the starting point of the path. This ensures that failed nodes will get N - equal to the dimensionality of the search space - opportunities to expand before they are pushed to the end of the priority queue.

[Algorithm 5.1](#) shows the implementation of our algorithm - *eRRT*. There are two basic differences from the previous RRT algorithms present in the literature. Previous algorithms focus on generating a search tree in the (high dimensional) configuration space and occasionally look for deterministic paths in the task space. In our case, the deterministic gradient steps in task space are evaluated first, and we resort to exploration only when a gradient step fails. This is a major source of gain in planning time. Secondly, even when exploration is made, we bias it toward the least energy nodes. Its implementation is shown in function `BiasedExplore()` of [Algorithm 5.2](#). We make a fixed number of exploration samples and select that sample which results in the lowest cost nearest neighbour in the current tree. This procedure biases the exploration to take place near the low cost nodes. We show in [Section 5.7](#) that other algorithms that perform un-biased exploration lead the search to high cost regions disregarding the demonstrations and hence resulting in lower quality paths.

Although gradient steps result in fast progress of the planner, these steps do not follow the Voronoi bias and hence may result in multiple redundant explorations of certain regions of state space. In order to avoid this, we perform a nearest neighbour query to ensure that gradient steps do not explore too close to any existing node (line 28 of [Algorithm 5.1](#)). In effect, tree refinements are

Algorithm 5.1 Energy-RRT Algorithm

```
1: function ERRT( $\mathbf{x}_s, \mathbf{x}^*$ )
2:    $T \leftarrow \text{EMPTYTREE}()$ ;
3:    $T.\text{INSERT}(\mathbf{x}_s)$ ;
4:
5:   while  $iter < maxiter$  do
6:      $\mathbf{x}_{best} \leftarrow T.\text{BESTNODE}()$ ;
7:      $\mathbf{x}_{grad} \leftarrow T.\text{ERRTGRADSTEP}(\mathbf{x}_{best})$ ;
8:     if  $\mathbf{x}_{grad} \neq \text{NULL}$  then
9:        $T.\text{INSERT}(\mathbf{x}_{grad})$ ;
10:      continue;
11:    end if
12:
13:     $T.\text{DECREASEPRIORITY}(\mathbf{x}_{best})$ ;
14:     $\mathbf{x}_{rand} \leftarrow T.\text{BIASEDEXPLORE}()$ ;
15:     $\mathbf{x}_{new} \leftarrow T.\text{EXTEND}(\mathbf{x}_{rand})$ ;
16:    if  $\text{ISFREE}(\mathbf{x}_{new})$  then
17:       $T.\text{INSERT}(\mathbf{x}_{new})$ ;
18:    end if
19:  end while
20:
21: end function
22:
23: function ERRTGRADSTEP( $\mathbf{x}_{best}$ )
24:    $\mathbf{x}_{grad} \leftarrow \mathbf{x}_{best} - \text{model}.\text{GRADIENT}(\mathbf{x}_{best})$ ;
25:    $cost \leftarrow \text{model}.\text{COST}(\mathbf{x}_{grad})$ ;
26:    $\mathbf{x}_{near} \leftarrow T.\text{NEAREST}(\mathbf{x}_{grad})$ ;
27:   if  $cost < \text{model}.\text{COST}(\mathbf{x}_b)$  then
28:     if  $\text{ISFREE}(\mathbf{x}_{grad}) \ \&\ \|\mathbf{x}_{near} - \mathbf{x}_{grad}\| < step$  then
29:       return  $\mathbf{x}_{grad}$ ;
30:     else
31:       return  $\text{NULL}$ ;
32:     end if
33:   end if
34: end function
```

performed only via the biased exploration function and fast and deterministic tree extensions into low energy regions are performed by the gradient steps (function `ErrtGradStep()` of [Algorithm 5.1](#)).

[Figure 5.1\(d\)](#) shows a typical search tree when local minimums are present in the energy map. Note that the exploration is triggered only in regions where local minimums were present. Once the search navigated out of those regions and a clear gradient path was available, fast deterministic steps were taken to reach the goal. [Figure 5.1\(a\)-\(c\)](#) respectively show the searches adopted by the basic RRT exploration which disregards the cost map, *hRRT* exploration which is extremely aggressive in adhering to the cost map and *GRRT* exploration where cost aggressiveness can be tuned using the temperature scheduling parameters. In contrast, *eRRT* bootstraps the exploration using the dimensionality of the state space and node expansion failures.

[Figure 5.4](#) shows, from left to right, the tree development process for *eRRT*. The planner starts with gradient steps until it detects a local minimum ([Figure](#)

Algorithm 5.2 Tree Helper Algorithms

```
1: function INSERT( $\mathbf{x}_{new}, cost$ )
2:    $kdtree.INSET(\mathbf{x}_{new})$ ;
3:    $cost \leftarrow model.COST(\mathbf{x}_{new})$ ;
4:    $priority\_queue.INSET(\mathbf{x}_{new}, cost)$ 
5:    $\alpha[\mathbf{x}_{new}] \leftarrow -cost$ ;
6: end function
7:
8: function BIASEDEXPLORE( )
9:    $min\_cost \leftarrow \infty$ ;
10:  for  $i = 1 : dim$  do
11:     $\mathbf{x}_{rand} \leftarrow RANDOMSTATE()$ ;
12:     $\mathbf{x}_{near} \leftarrow kdtree.NEAREST(\mathbf{x}_{rand})$ ;
13:     $cost \leftarrow model.COST(\mathbf{x}_{near})$ ;
14:    if  $cost < min\_cost$  then
15:       $cost \leftarrow min\_cost$ ;
16:       $\mathbf{x}_{result} \leftarrow \mathbf{x}_{rand}$ ;
17:    end if
18:  end for
19:  return  $\mathbf{x}_{result}$ ;
20: end function
21:
22: function BESTNODE( )
23:  return  $priority\_queue.FRONT()$ ;
24: end function
25:
26: function DECREASEPRIORITY( $\mathbf{x}_{fail}$ )
27:    $cost \leftarrow model.COST(\mathbf{x}_{fail})$ ;
28:    $cost \leftarrow cost * \sqrt[N]{h(\mathbf{x}_{start})/h(\mathbf{x}_{fail})}$ ;
29:    $priority\_queue.UPDATEPRIORITY(\mathbf{x}_{fail}, cost)$ ;
30: end function
```

5.4(a)) and no more gradient steps are possible. The planner then explores around the local minimum until it finds feasible gradient steps, and continues with the gradient steps thereafter.

5.6 Hybrid Planning with $eRRT$

In most applications the notion of a cost-map is tied to the task space. This is due to the fact that demonstrations are easier to provide in the (typically) lower dimensional task space. Whereas planning and obstacle avoidance must be performed in the higher dimensional configuration space. For instance, in the case of a N-DOF robotic manipulator we may want to specify the path of the end-effector via demonstrations. However, the planner must search for a collision free path in the joint space of the manipulator. In the literature, this problem is termed as *hybrid planning*. Many previous works combine RRT exploration in task space with different obstacle avoidance controllers in the joint space Shkolnik and Tedrake (2009); Behnisch et al. (2010). Although these approaches can leverage existing obstacle avoidance controllers, they are computationally slower due to explicit obstacle avoidance computation in configuration space. Their

Algorithm 5.3 Hybrid Energy-RRT Algorithm

```
1: function ERRTHYBRID( $\mathbf{q}_s, \mathbf{x}^*, model$ )
2:    $T \leftarrow \text{EMPTYHYBRIDTREE}()$ ;
3:    $\mathbf{x}_s \leftarrow \text{FWDKIN}(\mathbf{q}_s)$ ;
4:    $T.\text{INSERT}(\mathbf{q}_s, \mathbf{x}_s)$ ;
5:
6:   while  $iter < maxiter$  do
7:      $(\mathbf{q}_{best}, \mathbf{x}_{best}) \leftarrow T.\text{BESTNODE}()$ ;
8:      $\mathbf{x}_{grad} \leftarrow T.\text{ERRTGRADSTEP}(\mathbf{x}_{best})$ ;
9:     if  $\mathbf{x}_{grad} \neq \text{NULL}$  then
10:       $\mathbf{q}_{grad} \leftarrow \mathbf{q}_{best} + J^+(\mathbf{x}_{grad} - \mathbf{x}_{best})$ ;
11:       $T.\text{INSERT}(\mathbf{q}_{grad}, \mathbf{x}_{grad})$ ;
12:      continue;
13:    end if
14:
15:     $\mathbf{q}_{rand} \leftarrow T.\text{BIASEDCONFIGURATIONEXPLORE}()$ ;
16:     $\mathbf{q}_{new} \leftarrow T.\text{EXTEND}(\mathbf{q}_{rand})$ ;
17:    if  $\text{ISFREE}(\mathbf{q}_{new})$  then
18:       $\mathbf{x}_{new} \leftarrow \text{FWDKIN}(\mathbf{q}_{new})$ ;
19:       $T.\text{INSERT}(\mathbf{q}_{new}, \mathbf{x}_{new})$ ;
20:    end if
21:  end while
22: end function
```

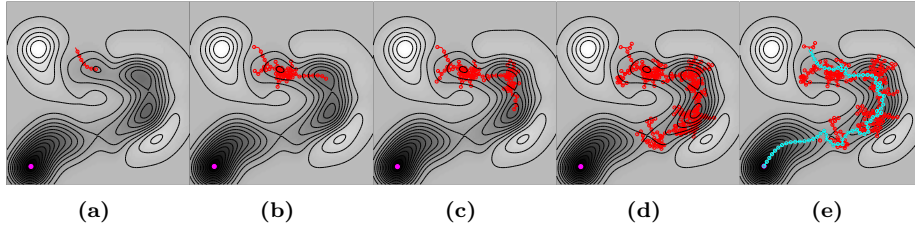


Figure 5.4: Progress of the *eRRT* algorithm. There are three local minima along the low-energy path to the goal. Fast deterministic (gradient) steps are taken until they hit a local minimum. Exploration is triggered around each of the local minima until the planner finds a feasible gradient direction.

performance varies depending on the relative configuration of the obstacles and the type of reactive controller used. Another body of works [Vahrenkamp et al. \(2009\)](#) relies on parallel exploration in the configuration space to find collision free nodes from which successful task space exploration can take place. A probabilistic perspective on hybrid planning was presented by [Toussaint \(2009\)](#). They cast the robot motion planning problem as one of probabilistic inference over a set of random variables. Interesting parallels were drawn between classical optimization based planning/control and message passing algorithms in [Toussaint and Goerick \(2010\)](#). They showed that these message passing algorithms equivalently solve the hybrid planning problem in the task and configuration spaces. Their approach was however based on local linearizations of an obstacle potential function and hence prone to local minima.

In this section, we will extend *eRRT* as a hybrid planner. We combine *eRRT* with the J^+RRT approach [Vahrenkamp et al. \(2009\)](#) for exploration in joint

space to avoid obstacles. Results in [Section 5.7](#) show that the computational advantage of $eRRT$ is also reflected in its hybrid counterpart, especially in more difficult planning problems. The speedup in the case of hybrid $eRRT$ again comes from exploiting the information from the energy function. The J^+RRT algorithm makes primary explorations in the configuration space, and with some probability $pGoal$, tries to take steps in the task space (using Jacobian pseudo-inverse) toward the goal. The computational cost hence depends on the parameter $pGoal$. For difficult problems it is computationally advantageous to set this parameter to a small value so that the planner explores more often. For easier problems it should be kept large since it is more likely that a direct path is available to the goal and the planner should check for it more frequently. In contrast, the $eRRT$ algorithm presented here makes primary explorations in the (lower dimensional) task space by taking gradient steps and using the Jacobian pseudo-inverse to compute corresponding configurations. Similar to the basic $eRRT$, in case of failure of a gradient step, we explore the configuration space using random exploration biased toward the best (in terms of energy) known configuration.

[Algorithm 5.3](#) explains the hybrid implementation of $eRRT$. The hybrid tree has small additions as compared to the basic $eRRT$. The nodes now contain both the configuration and task-space variables. When a gradient step is taken in the task space, the new configuration is computed using Jacobian pseudo-inverse. If a gradient step fails, the tree is extended using biased Voronoi exploration in the configuration space and the corresponding task-space variable is computed using forward-kinematics.

5.7 Results

In this section we first compare the basic (non-hybrid) $eRRT$ with three other algorithms: basic RRT , $hRRT$ and $GRRT$ with two different temperature schedules. All the implementations are done in MATLAB with very similar object-oriented structure in order for the results of different algorithms to be comparable. The planners were run on a machine with 2.2 GHz Intel i5 processor and 4 GB of RAM. We compare these algorithms on three different toy problems with varying complexity of demonstrations. Next, we compare the performance of our hybrid algorithm against J^+RRT with two different values of its $pGoal$ parameter on two toy problems with obstacles. The difficulty of the planning problem is varied incrementally in order to show that $eRRT$ is computationally more efficient especially in difficult problems. Finally we present a real robot experiment on the KUKA-LWR manipulator grasping an object in a cluttered environment. We also compare the performance of our hybrid algorithm with J^+RRT on a benchmark task where the manipulator has to move through a small window to the other side, while avoiding collisions.

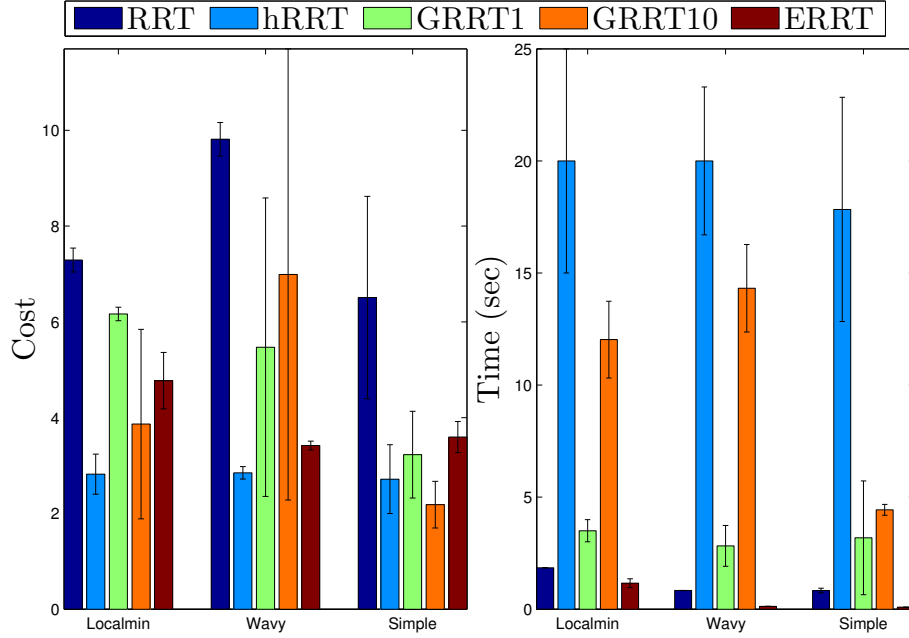


Figure 5.5: Performance evaluation for basic $eRRT$ on the three different toy examples with respect to cost of the obtained path (**left**) and time spent for planning (**right**).

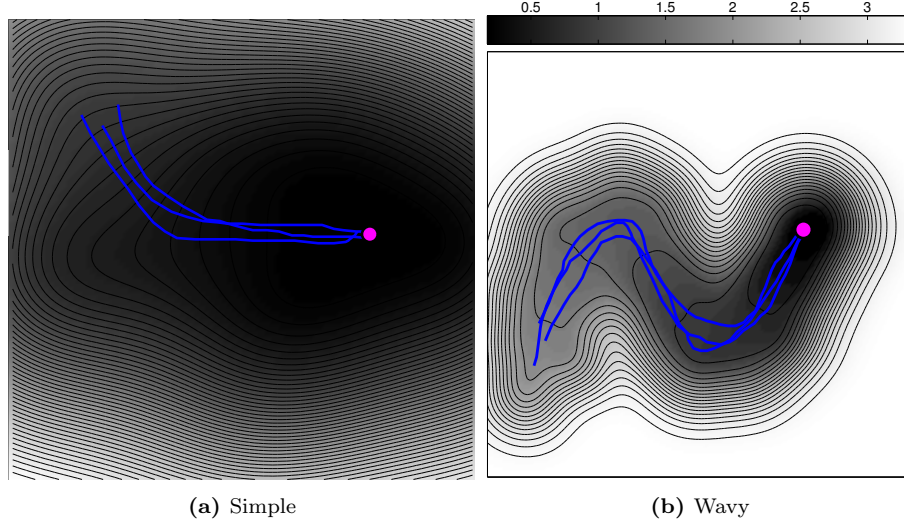


Figure 5.6: Demonstrations and their learned energy maps for the hybrid toy example. Blue lines show the demonstrations of how to reach the target. Contours of the energy function learned from these demonstrations are also shown.

5.7.1 EVALUATING THE BASIC $eRRT$

We use 2D demonstrations of varying complexity to learn the energy-map and use $eRRT$ to plan over them. Apart from the demonstrations shown in Figure 5.1(a), we use two more demonstration sets as shown in Figure 5.6. We term these problems *simple*, *wavy* and *localmin*. For comparison, we learn a cost-map

from the same demonstrations using the method presented in Berenson et al. (2011) and use it for planning with the other algorithms (*hRRT* and *GRRT*). Figure 5.5 shows the performance of all the algorithms with respect to the cost of the obtained path and the planning time. *GRRTn* refers to the algorithm *GRRT* being run with its temperature parameter (*nFailMax* in Berenson et al. (2011)) set to the value *n*. Error bars are computed over 5 different runs of the algorithms starting from random locations normally distributed around the demonstrations. We see from Figure 5.5 that the cost-agnostic basic *RRT* gives the most costly paths and *hRRT* gives the most cost-efficient paths. However, we also see from Figure 5.5 that *hRRT* is the slowest of all algorithms, which is due to its cost-aggressiveness. *GRRT* algorithms provide a trade-off between performance and path quality with the tunable temperature parameter. Larger the parameter, more cost-aggressive the algorithm. Hence, *GRRT1* is faster than *GRRT30* but gives lower quality paths. In comparison, *eRRT* gives moderate cost paths on all problems. In most cases the cost is of the same order as *GRRT1* and *GRRT30*. However, the planning time of *eRRT* is extremely small as compared to other algorithms. Note that *eRRT* does not need tuning of a temperature parameter according to the difficulty of the problem. This is due to the selective nature of exploration in the algorithm that spends more time exploring the space only if needed. This also explains the slightly more time spent by *eRRT* in the *localmin* problem than the other two. Many failures due to the local minima lead to higher exploration rate.

The energy function learned from demonstrations is more reliable near the demonstrations. I.e., far from demonstrations the gradients of the function are not optimized by the learning framework and hence can be arbitrary. However the model retains some generalization due to smoothness imposed by the kernel (rbf in these experiments). Here we test the generalizability of the learned model far from the demonstrations by increasing the variance with which the initial configuration is sampled around the demonstrations. We increase the variance upto 50% of the size of the state space and compute the planning time and cost of the trajectory obtained. Figure 5.7 shows the results. *eRRT* consistently gives moderate quality paths across the range of variances used. The only algorithms that give better quality paths (shown in blue and orange colors) spend an order of magnitude more time for planning.

5.7.2 EVALUATING THE HYBRID *eRRT*

We evaluate and compare the performance of the hybrid version of *eRRT* with two other algorithms. *TSRRT* - which builds the search tree in task-space and exploits an existing obstacle avoidance controller to explore the configuration space - and *J⁺RRT* - that explores the configuration space in the standard RRT fashion and occasionally searches the task space for a path to the goal.

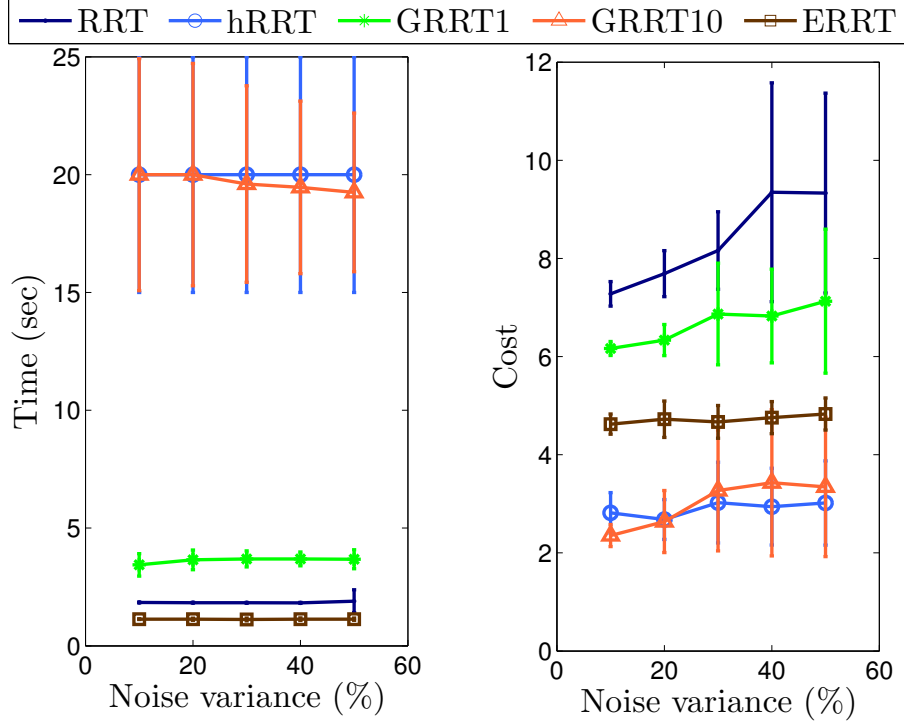


Figure 5.7: Evaluation of generalization ability of the learned energy-function for the *localmin* problem. Variance of the initial configuration sampler is indicated in terms of percentage span of the state space. There is no significant increase in cost of the paths obtained or in the computational effort as the starting configuration is chosen away from the demonstrations. Error bars are computed over 5 different samples for each value of the variance.

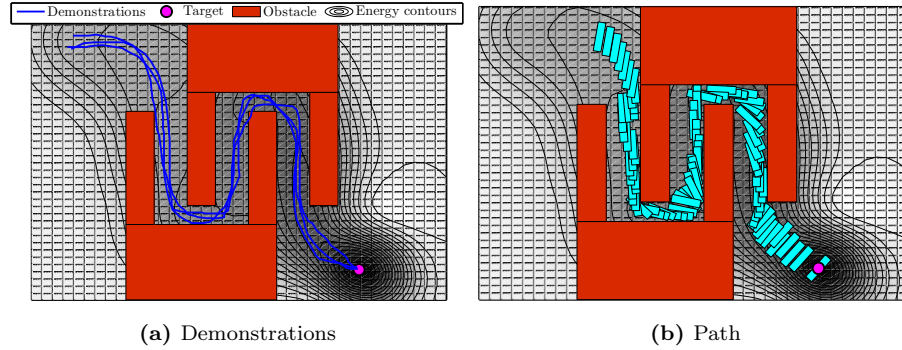


Figure 5.8: Toy setup for evaluating the hybrid planning algorithms. Energy map learned from the demonstrations assists the *eRRT* planner in finding a collision free path to the goal. The robot is planar with 3 DOF configuration space (shown in (b)).

TOY PROBLEMS

Figure 5.8(a) shows the 2D state space with two obstacles (shown in red) between the start and goal. A planar robot is required to move in a 3D configuration space - two dimensions representing the position of its center and one representing its orientation with respect to the positive X-axis. We consider the

Figure 5.9 Planning time for different hybrid planners. “*eRRT – OA*” denotes the *eRRT* algorithm with explicit obstacle avoidance. It uses the same obstacle avoidance algorithm as used in *TSRRT*. All the algorithms were given maximum time of 180sec. Note that in most runs *JRRT0.8* fails to find a path within this limit.

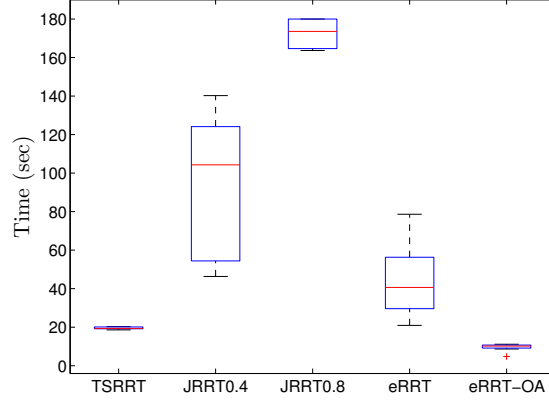
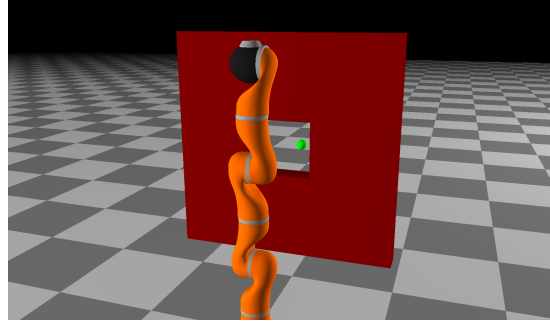


Figure 5.10 Setup for evaluating the hybrid planners. The planner is required to find a collision free path in configuration space to move toward the target (shown in green) on the other side of a small window.



2D position as the task space and provide demonstrations (shown in blue) on how to maneuver the center point of the robot to the goal. This setup mimics the fact that we usually provide demonstrations in the task space whereas path planning and obstacle avoidance depend on the full configuration. In this case, the robot must change its configuration (orientation) while passing through the obstacles since it can only pass through the top and bottom corridors if the orientation is close to horizontal. Figure 5.8(b) shows one such path obtained from our planner. Planning times of the different algorithms are shown in Figure 5.9. Error bars are computed over 10 trials starting with random initial configurations. *JRRT* algorithms’ performance depends on the parameter $pGoal$ which we set to 0.4 and 0.8. Although *eRRT* performs better than the *JRRT* algorithms, it is however surpassed by the *TSRRT* algorithm which uses an explicit obstacle avoidance. Since the configuration space here consists of only one extra dimension compared to the task space, performing obstacle avoidance is simply a line search. Hence, in this particular problem *TSRRT* gets the benefit of easy obstacle avoidance without the additional computational burden. As shown in Behnisch et al. (2010), in complex real-world scenarios, the use of explicit obstacle avoidance in *TSRRT* results in a slowdown. Nevertheless, Figure 5.9 shows that combining explicit obstacle avoidance with *eRRT* (denoted by “*eRRT – OA*”) results in even lower planning times. Hence, an efficient obstacle avoidance controller, if available, can be combined with *eRRT*. On the other hand, as is the case in most real-world problems, obstacle avoidance is compu-

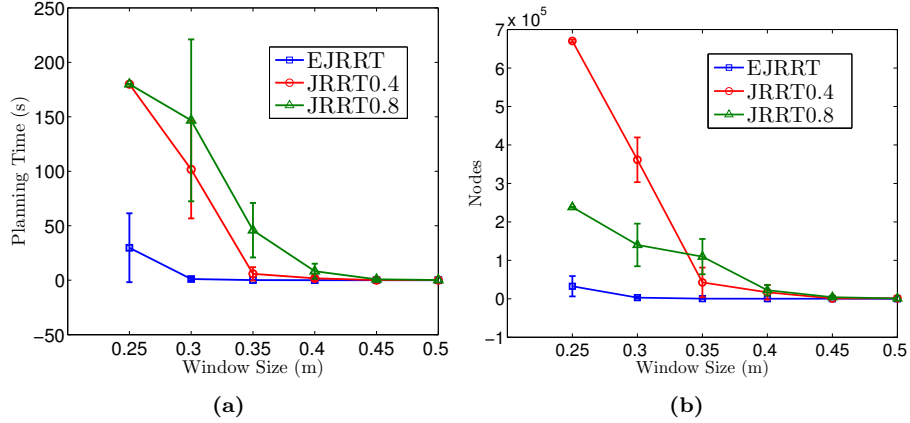


Figure 5.11: Planning time and number of nodes added for *eJRRRT* and *JRRRT* with two different parameters. With larger window size, the problem is easier to solve and hence the difference between the performances of different algorithms is minor. At window size 0.25, both *JRRRT* algorithms fail to find a path within 3 minutes.

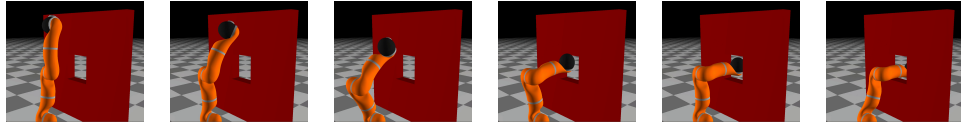


Figure 5.12: The robot moving through the window to reach the target on the other side. The size of the window shown here is 0.25m which is close to the robot's link width $\approx 0.15m$.

tationally demanding, *eJRRRT* may rely on exploration for obstacle avoidance. It is worth mentioning here that the main source of the better performance of *eJRRRT* is that the tree expansion is guided by the energy function. This creates a bias for newly sampled nodes in regions where they are most likely to be part of a successful path.

BENCHMARK PROBLEM

We use the 7-DOF KUKA-LWR in simulation and the real platform to evaluate our method on a benchmark problem. Figure 5.10 shows the setup where the planner is required to find a path in the joint-space of the manipulator from a starting configuration on one side of the window to a Cartesian location on the other side without colliding with the window frame. Demonstrations are provided on how to move the end-effector to the target Cartesian location and the energy function is learned. We use 3D position as the task space and plan paths with varying window sizes, thereby making the problem increasingly difficult. Figure 5.12 shows one of the obtained solutions implemented on the simulated robot. Figure 5.11 compares the planning times with different window sizes for *eJRRRT* and the *J⁺JRRRT* algorithms with different temperature parameters. Error bars are computed for 10 different runs starting from normally distributed configurations with mean configuration as shown in Figure 5.10 and a variance

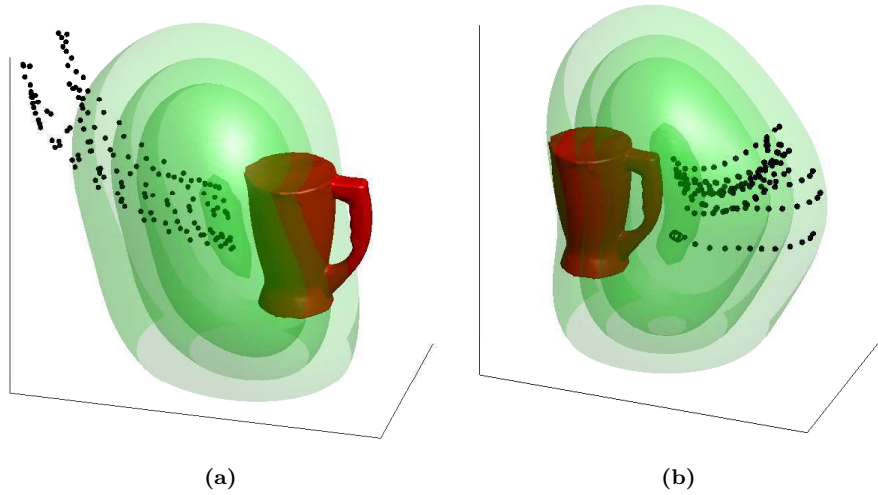


Figure 5.13: Energy function iso-surfaces in a 3D state space. Demonstrations that resulted in this energy function are shown as black dots. The contours depict local minima around two graspable parts of the object - one on the handle and the other on the opposite face. Running the planner on this energy map results in paths terminating at one of the two graspable patches.

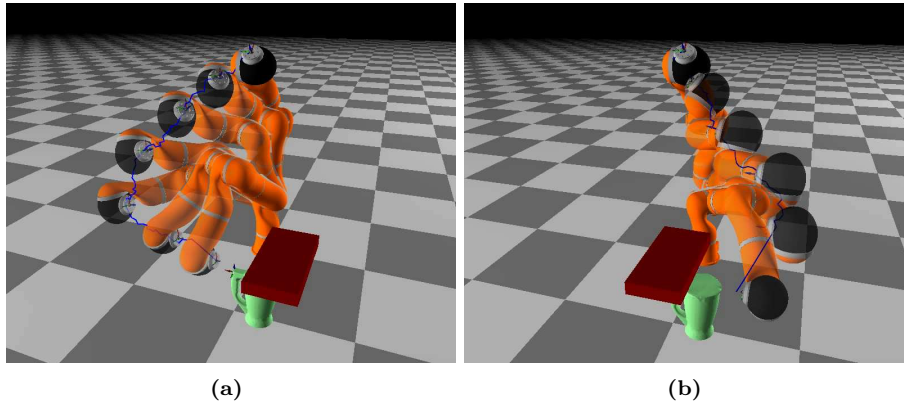


Figure 5.14: Two different paths obtained with slightly different configuration of the obstacle (shown in red). The planner chooses a goal configuration out of the two (handle and opposite face) goal regions as specified implicitly by the energy function.

of 20 degrees at each joint. Note that the difference in performance increases as the window size decreases. With window size of $0.25m$ both the *JRRT* algorithms fail to find a path within 3 minutes. This underlines the advantage of using the energy function in *eRRT* and the fact that it becomes more critical as the problem becomes harder to solve.

5.7.3 PLANNING WITH GOAL REGIONS

Here we present a real world example where we learn a continuous goal region from demonstrations. In this example, the goal region is a set of end-effector

poses that are feasible for grasping an object. Figure 5.13 shows an object with two graspable patches - one on the handle and other on the opposite face. The two patches consist of infinite number of end-effector positions that are acceptable for grasping the object. We employ hybrid eRRT to find a collision free path in joint-space for reaching to a graspable pose of the object. Figure 5.14 shows two such paths. Depending on how the object is placed, the planner automatically chooses one of the two graspable patches and finds a feasible path.

5.8 Conclusions

In this chapter we presented an approach for motion planning over energy maps. We departed from a purely local view of motion planning and presented an amalgamation of the local approaches presented in previous chapters with state-of-the-art global planning algorithms. Such a combination resulted in planning times that were significantly lower than the existing global planners, and at the same time, retain their probabilistic guarantees and the ability to handle arbitrary workspaces.

We compared our energy function method with the widely known cost-map approaches in literature and showed that the energy function is useful for planning in two important ways. It encodes more information than a cost map learned from the same set of demonstrations and hence improves the computational performance of randomized planning algorithms. Secondly, it can be used to implicitly encode multiple goals or goal-regions directly from demonstrations. We also presented the *eRRT* algorithm that exploits the properties of the learned energy-map to generate high quality paths faster than cost-map based algorithms presented in the literature. Our algorithm is different in spirit to the other cost-map planners in the fact that we trigger exploration only when it is required and is targeted toward low-energy regions of state space. We also extended the basic *eRRT* algorithm to a hybrid algorithm where the demonstrations (and hence the energy-map) are available in the task space and the planner operates in the configuration space. Results show that using an energy map results in a considerable gain in planning time.

The basic premise of our algorithm is that the demonstrations are *good* and the algorithm strongly relies on this assumption. A main drawback of the presented approach is that if the demonstrations are consistently moving away from the goal or deliberately moving toward the obstacles, it will lead the search to dead-ends and will result in large computational time. This also points to the fact that the computational gain in our algorithm is obtained by replacing the exhaustive search behaviour of RRT with a more targeted search. Secondly, the procedure for learning energy-map relies on open parameters (e.g. the kernel parameters). Although these are the usual parameters for any SVM based algorithm, they need to be tuned via cross-validation or grid search in the parameter space. Without proper model parameters the performance of the planner might

suffer as a good quality energy-map is critical to its performance.

CONCLUSIONS

6.1 Key Contributions

In this thesis we addressed several issues surrounding the problem of motion planning for the case of reach-to-grasp motions. While addressing specific problems in robotics, we also made contributions on a general algorithmic level.

6.1.1 ROBOTICS

From a robotics point of view, we identified the following key aspects of reach-to-grasp motions and proposed solutions to them.

ROBUST HAND-ARM COUPLING UNDER PERTURBATIONS

In the first part of this thesis, we followed a Programming by Demonstration approach to the problem of robust modelling of reach-grasp motions and took inspiration from the way humans adapt under perturbations. This was in contrast to previous works in motion modelling that use unperturbed motions for training a model and rely on the same for adapting under real-time perturbations. To endow a robot with this competence, we developed a Coupled Dynamical System based controller, whereby two dynamical systems driving the hand and finger motions were coupled using a non-linear coupling function derived from the demonstrations. This offers a compact encoding for reach-to-grasp motions which ensures fast adaptation with zero latency for re-planning.

CHOICE OF MULTIPLE GRASPING POINTS

Encoding robot motions using dynamical systems (DS) enables the robots to react against sudden perturbations. However, with only one attractor, the motions are always directed towards a single target. In [Chapter 3](#) we presented an approach for combining several such DS with distinct attractors, resulting in a multi-attractor DS. We showed its applicability in reach-to-grasp tasks where the attractors represent several grasping points on an object. While exploiting multiple attractors provides more flexibility in recovering from unseen perturbations, it also increases the complexity of the underlying learning problem. Here we presented the Augmented-SVM (A-SVM) model which inherited the region partitioning ability of the well known SVM classifier and was augmented with

novel constraints derived from the individual DS. Experiments were performed - in simulation and on the real 7 degree of freedom KUKA LWR arm - to establish the on-line replanning capability of our model.

GLOBAL PLANNING IN CONSTRAINED WORKSPACES

In the last part of this thesis we took a complementary view to motion planning, concentrating on global properties of the planning algorithm. Here we leveraged local information from demonstrations in the form of a energy-map to achieve significant gains in the computational effort required for global planning. The special properties of the energy-map made it possible to expedite a sampling based search by restricting exploration only to regions where it is needed and suggesting directions for search based on the demonstrations. We also presented a randomly exploring random tree based algorithm to search for low-energy paths over the learned map. In summary, state-of-the-art sampling based planning algorithms complemented the local techniques developed in the first part of this thesis by removing the problem of local minima. With our hybrid approach, although the cost of planning a collision free path increased as compared to a purely local approach, it remained considerably less than a purely global approach.

6.1.2 MACHINE LEARNING

From an algorithmic and machine learning standpoint, this thesis presented two novel contributions.

COUPLED DYNAMICAL SYSTEM

Although the CDS model presented in [Chapter 2](#) focusses on modelling hand-finger coordination, the framework is useful in many other scenarios with different types of couplings. In subsequent works, this framework has been used to model a) coupling between the different axes of motion in order to maintain a *direction of approach* for robust grasping ([Figueiredo et al., 2012](#)) b) coordination between more than two subsystems, e.g., eye-arm-hand coordination ([Lukic et al., 2014](#)) c) coordinated bimanual motions in (part of ongoing PhD thesis by Lucia Ureche). These different applications show that coupling information is critical in a variety of tasks and that our model is generally applicable to such tasks in its ability to encapsulate and reproduce the coupling information in real-time.

GRADIENT CONSTRAINED SVM

In [Chapter 4](#) we developed the Augmented-SVM formulation of [Chapter 3](#) to a general function approximation technique which can encapsulate simultaneous constraints on the function value as well as its gradient. Traditionally, the

SVM framework has been used for function approximation when function values (labels in classification, scalars in regression) at specific inputs are available for training. In our work, we re-formulated the classical SVM with gradient constraints using the KKT machinery and Lagrange duality. We show that the resulting dual is a larger quadratic program retaining the convexity properties of the classical SVM and derive globally convergent sequential-minimal-optimization (SMO) like iterates for solving it efficiently. We show that such a formulation is useful in two real-world applications, i.e., energy function learning and implicit surface reconstruction. These two apparently different problems were shown to be quite similar at the level of the learning formulation, i.e., they require similar constraints on the function value and gradient and hence both could be represented as special cases of our formulation. We also showed that our learning scheme leverages gradient information to statistically reduce the testing errors.

6.2 Limitations and Future Work

Throughout this thesis, we have maintained a common application theme for reach-to-grasp motions in a way that subsequent chapters attempt to address some of the shortcomings of the approaches presented in previous chapters. However each of the individual methods have drawbacks that may be addressed by independent research directions. Here we discuss these limitations and propose future work.

6.2.1 GENERALIZED COUPLING IN CDS MODELS

In our presentation of the CDS model in [Chapter 2](#), we assumed that the master-slave variables and the coupling function are known prior to the modelling. Although this was the case in hand-finger coupling, the coupling might not be explicitly known in more elaborate and complex task spaces. In such cases automatic coupling detection should be employed in order to bootstrap the modelling. A wrong choice of the master and slave variables might result in poor task performance even with a good fit between the data and the learned model.

Secondly, the nature of the coupling assumed in our model was only unidirectional, i.e., a perturbation in the reaching motion was reflected in the finger motion and not the other way round. A typical case where such a bi-directional coupling would be useful is when the dynamic controller for the fingers is malfunctioning and is too slow to follow the desired trajectory. In such a scenario, the arm must also slow down and synchronize with the finger motion. Such a behaviour, even if not biologically inspired, is desirable in the context of robotics where controller noise is ubiquitous. However, it needs to be studied what effect (in terms of stability) this bi-directional dependency will have on the overall system.

6.2.2 GLOBAL STABILITY IN MULTIPLE-ATTRACTOR DS

The extension of single attractor DS to one with multiple attractors increased the robustness of reaching motions under perturbations. However, being a completely data driven approach it lacked the guarantee of global asymptotic stability available in the single attractor case. In effect, we trade-off the global asymptotic stability with the possibility to model a richer variety of motions and with multiple attractors. Further work is required to characterize the number and locations of spurious attractors that emerge in the absence of global stability constraints. The characterization may be sought in terms of the type of kernels or their parameters (such as the kernel width in the rbf kernel). Such a study would be useful in either ensuring that no spurious attractors exist, or that they all lie outside the region of interest for the task.

6.2.3 INCREMENTAL GLOBAL PLANNING

We propose two major directions of work branching from our global planning framework of [Chapter 5](#). The methodology presented there aimed at combining local information from demonstrations with global properties of sampling based planners. A major desirable property of local planning approaches is the ability to react against changes in the environment without resorting to a full replanning step. The algorithm presented in [Chapter 5](#) lacks this property. A promising direction for future work is to enhance our algorithm to re-use previous searches when replanning is required. This would also allow a closed-loop implementation and ensure continuity between paths returned by successive calls to the planner.

Secondly, the biased sampling method employed in our algorithm is the bottleneck for the computation time. This is because a number of explorations are made (and rejected) without any progress to the planner. We use this approach in order to maintain both the low-energy and Voronoi biases in the explorations. However, maintaining an estimate of the Voronoi regions for each node would help the planner take an exploratory step without performing a number of (rejected) samplings. Constructing Voronoi regions is however computationally prohibitive to perform at every iteration. Estimating the Voronoi regions incrementally as the search tree grows would result in a significantly faster implementation of the same algorithm as presented in this thesis.

6.2.4 ROBOT DYNAMICS CONSIDERATION

The overall vision and the work presented in this thesis concentrates on the first order dynamics of the task and does not take into account the dynamic properties of the robot. In our implementations we have decoupled the two problems by using Jacobian based inverse-kinematics and inverse-dynamics to compute joint angle commands to the robots. This approach is inefficient since the desired task space trajectories may not be optimal, and in some extreme cases, might not be feasible given the dynamic constraints of the robot. A potential

direction to address this issue may be to include the robot's dynamic equations and constraints while computing task models from demonstrations. Formally, such problems can be formulated in the optimal control framework. Such an approach will also allow to strike a balance between the two measures of optimality: a) closely following the demonstrations and b) robot-specific dynamics. In effect, this will allow one to tune the task model differently according to the capabilities of each robot.

Appendices

APPENDIX

A.1 Stability of CDS model

To prove that the CDS model indeed follows the conditions 2.9, we use the properties of its individual components. For simplicity, we shift all the data into the goal reference frame so that $\xi_x^* = \xi_f^* = 0$. The condition 2.9a holds true due to the global stability of SEDS. To investigate the stability of the coupling, we consider

$$\begin{aligned}
 \lim_{t \rightarrow \infty} \mathbb{E} \left[\tilde{\xi}_f | \Psi(\xi_x) \right] &= \mathbb{E} \left[\tilde{\xi}_f | \Psi \left(\lim_{t \rightarrow \infty} \xi_x \right) \right] \\
 &= \mathbb{E} \left[\tilde{\xi}_f \middle| \lim_{\xi_x \rightarrow \mathbf{0}} \Psi(\xi_x) \right] \text{ (By 2.9a)} \\
 &= \mathbb{E} \left[\tilde{\xi}_f | \mathbf{0} \right] \text{ (By 2.7)} \\
 &= \mathbf{0} \text{ (By 2.8)}
 \end{aligned} \tag{A.1}$$

The model which governs the evolution of the coupled variable ξ_f is given by

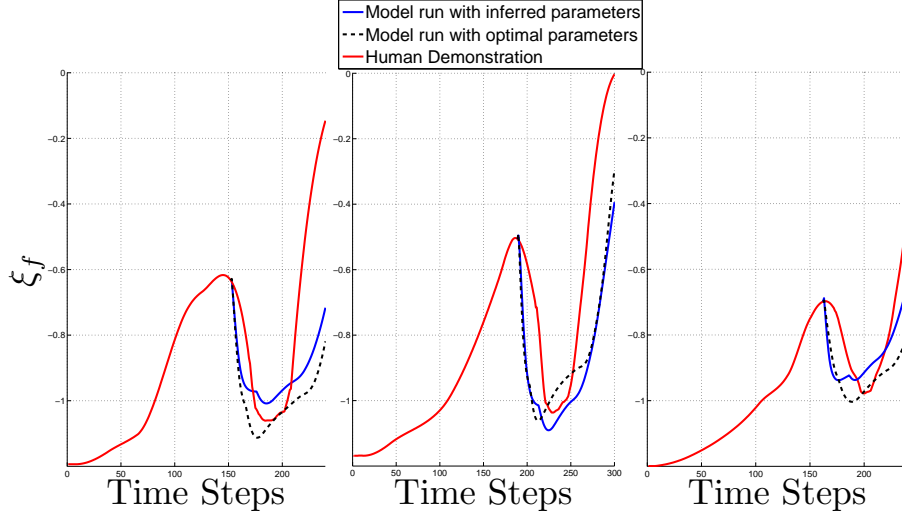
$$\dot{\xi}_f = \mathbb{E} \left[\dot{\xi}_f \middle| \left(\xi_f - \tilde{\xi}_f \right) \beta \right].$$

Taking the limiting values and using A.1, we get

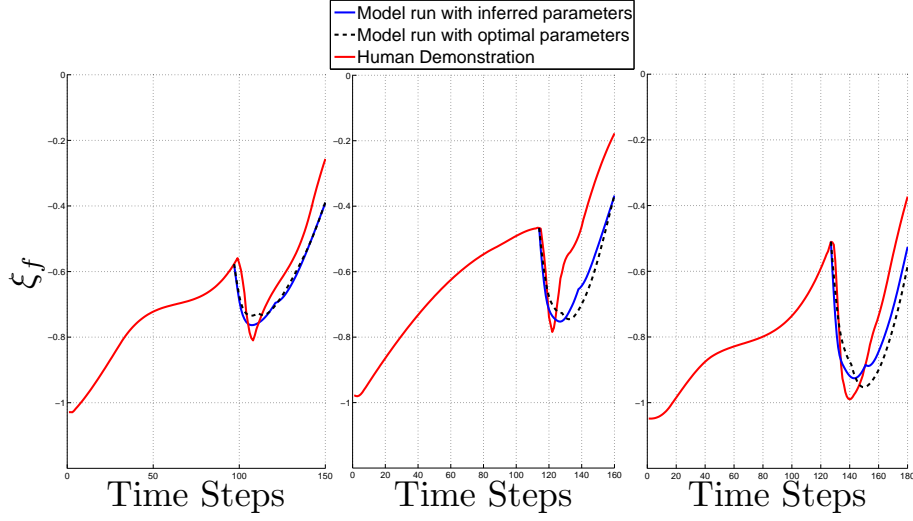
$$\lim_{t \rightarrow \infty} \dot{\xi}_f = \mathbb{E} \left[\dot{\xi}_f | \beta \xi_f \right]$$

which is again globally asymptotically stable due to SEDS. However, as seen from Algorithm 2.1, the multiplier α boosts the velocity before incrementing the state. It is trivial to see that this does not affect the global asymptotic behavior of the model since negative definite $\frac{A+A^T}{2} \Rightarrow \alpha \frac{A+A^T}{2}$ is also negative definite for $\alpha > 0$. For details on why such a condition is required for global stability, the reader is referred to Khansari Zadeh and Billard (2010b).

A.1.1 CDS MODEL PERFORMANCE WITH INFERRED PARAMETERS



(a) Subject 2



(b) Subject 3

Figure A.1: Comparison of model run with inferred parameter values, optimal parameter values and the actual human demonstrations under perturbation.

A.2 Kernel Derivatives

For scalar variables $x_i, x_j \in \mathbb{R}$ and any feature transformation $\phi : \mathbb{R} \mapsto \mathbb{R}^F$ we define a valid Mercer kernel as $k(x_i, x_j) \equiv \phi(x_i)^T \phi(x_j)$. If \prime denotes the derivative w.r.t the state variable, then the identities $\phi'(x_i)^T \phi(x_j) = \frac{\partial k(x_i, x_j)}{\partial x_i}$ and $\phi'(x_i)^T \phi'(x_j) = \frac{\partial^2 k(x_i, x_j)}{\partial x_i \partial x_j}$ follow directly from the definition of the kernel. We can rewrite these identities for vector variables $\mathbf{x}_i, \mathbf{x}_j \in \mathbb{R}^N$ by taking the derivative w.r.t one of the components (say n -th) as $\left(\frac{\partial \phi(\mathbf{x}_i)}{\partial \mathbf{x}(n)} \right)^T \phi(\mathbf{x}_j) = \frac{\partial k(\mathbf{x}_i, \mathbf{x}_j)}{\partial \mathbf{x}_i(n)}$. Expanding the first vector term we get

$$\Rightarrow \left[\frac{\partial \phi_1(\mathbf{x}_i)}{\partial \mathbf{x}(n)}, \frac{\partial \phi_2(\mathbf{x}_i)}{\partial \mathbf{x}(n)}, \dots, \frac{\partial \phi_F(\mathbf{x}_i)}{\partial \mathbf{x}(n)} \right] \phi(\mathbf{x}_j) = \frac{\partial k(\mathbf{x}_i, \mathbf{x}_j)}{\partial \mathbf{x}_i(n)}.$$

Stacking the above equation in rows for $n = 1 \dots N$, we get

$$\begin{bmatrix} \frac{\partial \phi_1(\mathbf{x}_i)}{\partial \mathbf{x}(1)} & \frac{\partial \phi_2(\mathbf{x}_i)}{\partial \mathbf{x}(1)} & \dots & \frac{\partial \phi_F(\mathbf{x}_i)}{\partial \mathbf{x}(1)} \\ \frac{\partial \phi_1(\mathbf{x}_i)}{\partial \mathbf{x}(2)} & \frac{\partial \phi_2(\mathbf{x}_i)}{\partial \mathbf{x}(2)} & \dots & \frac{\partial \phi_F(\mathbf{x}_i)}{\partial \mathbf{x}(2)} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \phi_1(\mathbf{x}_i)}{\partial \mathbf{x}(N)} & \frac{\partial \phi_2(\mathbf{x}_i)}{\partial \mathbf{x}(N)} & \dots & \frac{\partial \phi_F(\mathbf{x}_i)}{\partial \mathbf{x}(N)} \end{bmatrix} \phi(\mathbf{x}_j) = \begin{bmatrix} \frac{\partial k(\mathbf{x}_i, \mathbf{x}_j)}{\partial \mathbf{x}_i(1)} \\ \frac{\partial k(\mathbf{x}_i, \mathbf{x}_j)}{\partial \mathbf{x}_i(2)} \\ \vdots \\ \frac{\partial k(\mathbf{x}_i, \mathbf{x}_j)}{\partial \mathbf{x}_i(N)} \end{bmatrix}$$

$$\Rightarrow \mathbf{J}(\mathbf{x}_i)^T \phi(\mathbf{x}_j) = \frac{\partial k(\mathbf{x}_i, \mathbf{x}_j)}{\partial \mathbf{x}_i} \quad (\text{A.2})$$

where \mathbf{J} denotes the standard Jacobian matrix for a vector valued function. Similarly, by writing the derivatives w.r.t (n, m) -th dimension and putting them as the corresponding element of a Hessian matrix we get

$$\mathbf{J}(\mathbf{x}_i)^T \mathbf{J}(\mathbf{x}_j) = \frac{\partial^2 k(\mathbf{x}_i, \mathbf{x}_j)}{\partial \mathbf{x}_i \partial \mathbf{x}_j}. \quad (\text{A.3})$$

A.3 Specific kernel expansions

The above formulation is generic and can be applied to any kernel. Here we give the rbf kernel specific expressions for the block matrices in 3.15.

A.3.1 RBF KERNEL

$$\begin{aligned} [\mathbf{K}]_{ij} &= y_i y_j k(\mathbf{x}_i, \mathbf{x}_j) = y_i y_j e^{-d\|\mathbf{x}_i - \mathbf{x}_j\|^2} \\ [\mathbf{G}]_{ij} &= y_i \left(\frac{\partial k(\mathbf{x}_i, \mathbf{x}_j)}{\partial \mathbf{x}_j} \right)^T \hat{\mathbf{x}}_j = -2dy_i e^{-d\|\mathbf{x}_i - \mathbf{x}_j\|^2} (\mathbf{x}_j - \mathbf{x}_i)^T \hat{\mathbf{x}}_j \end{aligned}$$

Replacing \mathbf{x}_j by \mathbf{x}^* in the above equation we get

$$\begin{aligned} [\mathbf{G}_*]_{ij} &= y_i \left(\frac{\partial k(\mathbf{x}_i, \mathbf{x}^*)}{\partial \mathbf{x}^*} \right)^T \mathbf{e}_j = -2dy_i e^{-d\|\mathbf{x}_i - \mathbf{x}^*\|^2} (\mathbf{x}^* - \mathbf{x}_i)^T \mathbf{e}_j \\ [\mathbf{H}]_{ij} &= \hat{\mathbf{x}}_i^T \frac{\partial^2 k(\mathbf{x}_i, \mathbf{x}_j)}{\partial \mathbf{x}_i \partial \mathbf{x}_j} \hat{\mathbf{x}}_j = \hat{\mathbf{x}}_i^T \left[\frac{\partial}{\partial \mathbf{x}_i} \left\{ -2de^{-d\|\mathbf{x}_i - \mathbf{x}_j\|^2} (\mathbf{x}_j - \mathbf{x}_i) \right\} \right] \hat{\mathbf{x}}_j \\ &= 2de^{-d\|\mathbf{x}_i - \mathbf{x}_j\|^2} \left[\hat{\mathbf{x}}_i^T \hat{\mathbf{x}}_j - 2d \left\{ \hat{\mathbf{x}}_i^T (\mathbf{x}_i - \mathbf{x}_j) \right\} \left\{ (\mathbf{x}_i - \mathbf{x}_j)^T \hat{\mathbf{x}}_j \right\} \right]. \end{aligned}$$

Again, replacing \mathbf{x}_j by \mathbf{x}^* ,

$$[\mathbf{H}_*]_{ij} = \hat{\mathbf{x}}_i^T \frac{\partial^2 k(\mathbf{x}_i, \mathbf{x}^*)}{\partial \mathbf{x}_i \partial \mathbf{x}^*} \mathbf{e}_j = 2de^{-d\|\mathbf{x}_i - \mathbf{x}^*\|^2} \left[\hat{\mathbf{x}}_i^T \mathbf{e}_j - 2d \left\{ \hat{\mathbf{x}}_i^T (\mathbf{x}_i - \mathbf{x}^*) \right\} \left\{ (\mathbf{x}_i - \mathbf{x}^*)^T \mathbf{e}_j \right\} \right].$$

Replacing \mathbf{x}_i also by \mathbf{x}^* ,

$$[\mathbf{H}_{**}]_{ij} = \mathbf{e}_i^T \frac{\partial^2 k(\mathbf{x}^*, \mathbf{x}^*)}{\partial \mathbf{x}^* \partial \mathbf{x}^*} \mathbf{e}_j = 2d (\mathbf{e}_i^T \mathbf{e}_j).$$

A.3.2 POLYNOMIAL KERNEL

$$\begin{aligned} [\mathbf{K}]_{ij} &= y_i y_j k(\mathbf{x}_i, \mathbf{x}_j) = y_i y_j (\mathbf{x}_i^T \mathbf{x}_j + 1)^d \\ [\mathbf{G}]_{ij} &= y_i \left(\frac{\partial k(\mathbf{x}_i, \mathbf{x}_j)}{\partial \mathbf{x}_j} \right)^T \hat{\mathbf{x}}_j = y_i d (\mathbf{x}_i^T \mathbf{x}_j + 1)^{d-1} \mathbf{x}_i^T \hat{\mathbf{x}}_j. \end{aligned}$$

Replacing \mathbf{x}_j by \mathbf{x}_* in the above equation we get

$$[\mathbf{G}_*]_{ij} = y_i \left(\frac{\partial k(\mathbf{x}_i, \mathbf{x}_*)}{\partial \mathbf{x}_*} \right)^T \mathbf{e}_j = y_i d (\mathbf{x}_i^T \mathbf{x}_* + 1)^{d-1} \mathbf{x}_i^T \mathbf{e}_j.$$

$$\begin{aligned} [\mathbf{H}]_{ij} &= \hat{\mathbf{x}}_i^T \frac{\partial^2 k(\mathbf{x}_i, \mathbf{x}_j)}{\partial \mathbf{x}_i \partial \mathbf{x}_j} \hat{\mathbf{x}}_j \\ &= \hat{\mathbf{x}}_i^T \left[\frac{\partial}{\partial \mathbf{x}_i} \{d(\mathbf{x}_i^T \mathbf{x}_j + 1)^{d-1} \mathbf{x}_i\} \right] \hat{\mathbf{x}}_j \\ &= \hat{\mathbf{x}}_i^T \left[d(\mathbf{x}_i^T \mathbf{x}_j + 1)^{d-1} \frac{\partial \mathbf{x}_i}{\partial \mathbf{x}_i} + \mathbf{x}_i \frac{\partial}{\partial \mathbf{x}_i} \{d(\mathbf{x}_i^T \mathbf{x}_j + 1)^{d-1}\} \right] \hat{\mathbf{x}}_j \\ &= \hat{\mathbf{x}}_i^T [d(\mathbf{x}_i^T \mathbf{x}_j + 1)^{d-1} \mathbf{I}_N + d(d-1)(\mathbf{x}_i^T \mathbf{x}_j + 1)^{d-2} \mathbf{x}_i \mathbf{x}_j^T] \hat{\mathbf{x}}_j \\ &= d(\mathbf{x}_i^T \mathbf{x}_j + 1)^{d-2} \left[(\mathbf{x}_i^T \mathbf{x}_j + 1) \hat{\mathbf{x}}_i^T \hat{\mathbf{x}}_j + (d-1) (\hat{\mathbf{x}}_i^T \mathbf{x}_i) (\mathbf{x}_j^T \hat{\mathbf{x}}_j) \right]. \end{aligned}$$

Again, replacing \mathbf{x}_j by \mathbf{x}_* ,

$$\begin{aligned} [\mathbf{H}_*]_{ij} &= \hat{\mathbf{x}}_i^T \frac{\partial^2 k(\mathbf{x}_i, \mathbf{x}_*)}{\partial \mathbf{x}_i \partial \mathbf{x}_*} \mathbf{e}_j \\ &= d(\mathbf{x}_i^T \mathbf{x}_* + 1)^{d-2} \left[(\mathbf{x}_i^T \mathbf{x}_* + 1) \hat{\mathbf{x}}_i^T \mathbf{e}_j + (d-1) (\hat{\mathbf{x}}_i^T \mathbf{x}_i) (\mathbf{x}_*^T \mathbf{e}_j) \right]. \end{aligned}$$

Replacing \mathbf{x}_i also by \mathbf{x}_* ,

$$\begin{aligned} [\mathbf{H}_{**}]_{ij} &= \mathbf{e}_i^T \frac{\partial^2 k(\mathbf{x}_*, \mathbf{x}_*)}{\partial \mathbf{x}_*^2} \mathbf{e}_j \\ &= d(\mathbf{x}_*^T \mathbf{x}_* + 1)^{d-2} \left[(\mathbf{x}_*^T \mathbf{x}_* + 1) \mathbf{e}_i^T \mathbf{e}_j + (d-1) (\mathbf{e}_i^T \mathbf{x}_*) (\mathbf{x}_*^T \mathbf{e}_j) \right]. \end{aligned}$$

REFERENCES

- Pieter Abbeel and Andrew Y Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, page 1. ACM, 2004. [5.2](#)
- Russell L Andersson. Aggressive trajectory generator for a robot ping-pong player. *Control Systems Magazine, IEEE*, 9(2):15–21, 1989. [1.2.2](#)
- Brenna D Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robotics and autonomous systems*, 57(5):469–483, 2009. [1.2.2](#)
- Christopher G Atkeson, Andrew W Moore, and Stefan Schaal. Locally weighted learning for control. In *Lazy learning*, pages 75–113. Springer, 1997. [1.2.2](#), [1.2.2](#)
- Ji-Hun Bae, Suguru Arimoto, Yuuichi Yamamoto, Hiroe Hashiguchi, and Masahiro Sekimoto. Reaching to grasp and preshaping of multi-dofs robotic hand-arm systems using approximate configuration of objects. In *Intelligent Robots and Systems. IEEE/RSJ International Conference on*, volume 14, pages 1605–1610. IEEE, 2006. doi: 10.1109/IROS.2006.282050. [2.3.1](#)
- Sungchan Bae and Thomas J. Armstrong. A finger motion model for reach and grasp. *International Journal of Industrial Ergonomics*, 41(1):79 – 89, 2011. ISSN 0169-8141. doi: DOI:10.1016/j.ergon.2010.11.001. [2.3.2](#)
- Jerome Barraquand and Jean-Claude Latombe. Robot motion planning: A distributed representation approach. *The International Journal of Robotics Research*, 10(6):628–649, 1991. [1.2.1](#)
- Jerome Barraquand, Bruno Langlois, and J-C Latombe. Numerical potential field techniques for robot path planning. *Systems, Man and Cybernetics, IEEE Transactions on*, 22(2):224–241, 1992. [1.2.1](#)
- Matthias Behnisch, Robert Haschke, and Michael Gienger. Task space motion planning using reactive control. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 5934–5940. IEEE, 2010. [5.6](#), [5.7.2](#)
- Hamid R Berenji and Pratap Khedkar. Learning and tuning fuzzy logic controllers through reinforcements. *Neural Networks, IEEE Transactions on*, 3(5):724–740, 1992. [1.2.2](#)
- Dmitry Berenson, Siddhartha S Srinivasa, Dave Ferguson, Alvaro Collet, and James J Kuffner. Manipulation planning with workspace goal regions. In *Robotics and Automation, 2009. ICRA’09. IEEE International Conference on*, pages 618–624. IEEE, 2009. [2.2](#), [2.3.1](#), [5.2](#)

- Dmitry Berenson, Thierry Siméon, and Siddhartha S Srinivasa. Addressing cost-space chasms in manipulation planning. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 4561–4568. IEEE, 2011. [5.3](#), [5.7.1](#)
- A. Billard, S. Calinon, R. Dillmann, and S. Schaal. Survey: Robot programming by demonstration. In *Handbook of Robotics. Chapter 59*. MIT Press, 2008. [2.2](#)
- G.M. Bone, A. Lambert, and M. Edwards. Automated modeling and robotic grasping of unknown three-dimensional objects. In *Robotics and Automation. ICRA. IEEE International Conference on*, pages 292–298. IEEE, May 2008. doi: 10.1109/ROBOT.2008.4543223. [2.2](#)
- C. Bowen, G. Ye, and R. Alterovitz. Asymptotically optimal motion planning for learned tasks using time-dependent cost maps. *Automation Science and Engineering, IEEE Transactions on*, PP(99):1–12, 2014. ISSN 1545-5955. [5.3](#)
- Oliver Brock, James Kuffner, and Jing Xiao. Motion for manipulation tasks. *Springer Handbook of Robotics*, pages 615–645, 2008. [1.2.1](#)
- D. Bullock and S. Grossberg. The VITE model: A neural command circuit for generating arm and articulator trajectories. *Dynamic patterns in complex systems*, pages 305–326, 1988a. [2.3.2](#)
- Daniel Bullock and Stephen Grossberg. Neural dynamics of planned arm movements: emergent invariants and speed-accuracy properties during trajectory formation. *Psychological review*, 95(1):49, 1988b. [1.2.2](#)
- Brendan Burns and Oliver Brock. Sampling-based motion planning using predictive models. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 3120–3125. IEEE, 2005a. [5.3](#)
- Brendan Burns and Oliver Brock. Single-query entropy-guided path planning. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 2124–2129. IEEE, 2005b. [5.3](#)
- Brendan Burns and Oliver Brock. Toward optimal configuration space sampling. In *Robotics: Science and Systems*, pages 105–112. Citeseer, 2005c. [5.3](#)
- S. Calinon, F. Guenter, and A. Billard. On learning, representing, and generalizing a task in a humanoid robot. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 37(2):286–298, 2007. ISSN 1083-4419. [1.2.2](#)
- Sylvain Calinon. Robot programming by demonstration. In *Springer handbook of robotics*, pages 1371–1394. Springer, 2008. [1.2.2](#)
- Sylvain Calinon and Aude Billard. Recognition and reproduction of gestures using a probabilistic framework combining pca, ica and hmm. In *Proceedings of the 22nd international conference on Machine learning*, pages 105–112. ACM, 2005. [1.2.2](#)
- Sylvain Calinon and Aude Billard. Incremental learning of gestures by imitation in a humanoid robot. In *Proceedings of the ACM/IEEE international conference on Human-robot interaction*, pages 255–262. ACM, 2007. [1.2.2](#)

- Sylvain Calinon and Aude Billard. A probabilistic programming by demonstration framework handling constraints in joint space and task space. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 367–372. IEEE, 2008. [5.2](#)
- John Canny. *The complexity of robot motion planning*. MIT press, 1988. [1.2.1](#)
- U. Castiello, KMB Bennett, and GE Stelmach. Reach to grasp: the natural response to perturbation of object size. *Experimental Brain Research*, 94(1): 163–178, 1993. ISSN 0014-4819. [2.2](#)
- U. Castiello, K. Bennett, and H. Chambers. Reach to grasp: the response to a simultaneous perturbation of object position and size. *Experimental Brain Research*, 120(1):31–40, 1998. ISSN 0014-4819. [2.2](#), [2.3.2](#)
- Faïcel Chamroukhi, Allou Samé, Gérard Govaert, and Patrice Aknin. Time series modeling by a regression approach based on a latent process. *arXiv preprint arXiv:1312.6969*, 2013. [1.2.2](#)
- H.D. Chiang and C.C. Chu. A systematic search method for obtaining multiple local optimal solutions of nonlinear programming problems. *Circuits and Systems I: Fundamental Theory and Applications, IEEE Transactions on*, 43(2):99–109, 1996. [3.3](#)
- M. Christel and A. Billard. How monkeys morphology constrain natural prehension kinematics. Unconstrained conditions and simulation. *Behavioural brain research*, 131(1-2):169–184, 2001. [2.2](#)
- L. Dan and E. Todorov. Hierarchical optimal control of a 7-dof arm model. In *Adaptive Dynamic Programming and Reinforcement Learning, 2009. ADPRL '09. IEEE Symposium on*, pages 50 –57. IEEE, 2009. doi: 10.1109/ADPRL.2009.4927525. [2.2](#)
- Didier Devaurs, Thierry Siméon, and Juan Cortés. Enhancing the transition-based rrt to deal with complex cost spaces. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 4120–4125. IEEE, 2013. [5.3](#)
- K.R. Dixon and P.K. Khosla. Trajectory representation using sequenced linear dynamical systems. In *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, volume 4, pages 3925–3930. IEEE, 2004. [1.2.2](#), [3.1](#)
- Stanimir Dragiev, Marc Toussaint, and Michael Gienger. Gaussian process implicit surfaces for shape estimation and grasping. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 2845–2850. IEEE, 2011. [4.2](#)
- VT Elanayar, Yung C Shin, et al. Radial basis function neural network for approximation and estimation of nonlinear stochastic dynamic systems. *Neural Networks, IEEE Transactions on*, 5(4):594–603, 1994. [1.2.2](#)
- L.P. Ellekilde and H.I. Christensen. Control of mobile manipulator using the dynamical systems approach. In *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, pages 1370–1376. IEEE, 2009. [3.1](#)
- D.A. Engstrom and J.A.S. Kelso. Coordination dynamics of the complementary nature. *Gestalt theory*, 30:121, 2008. [2.2](#)

- Clemens Eppner, JÄijrgen Sturm, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard. Imitation learning with generalized task descriptions. In *ICRA '09*, pages 3968–3974. IEEE, 2009. [2.3.3](#)
- Alan Ettlin and Hannes Bleuler. Randomised rough-terrain robot motion planning. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 5798–5803. IEEE, 2006a. [5.3](#)
- Alan Ettlin and Hannes Bleuler. Rough-terrain robot motion planning based on obstacleness. In *Control, Automation, Robotics and Vision, 2006. ICARCV'06. 9th International Conference on*, pages 1–6. IEEE, 2006b. [5.3](#)
- Rui Figueiredo, Ashwini Shukla, Duarte Aragao, Plinio Moreno, Alexandre Bernardino, José Santos-Victor, and Aude Billard. Reaching and grasping kitchenware objects. In *International Symposium on System Integration*, 2012. [6.1.2](#)
- A. Fuchs and H. Haken. Pattern recognition and associative memory as dynamical processes in a synergetic system. i. translational invariance, selective attention, and decomposition of scenes. *Biol. Cybern.*, 60:17–22, November 1988. ISSN 0340-1200. [3.2](#)
- Maxim Garber and Ming C Lin. Constraint-based motion planning using voronoi diagrams. In *Algorithmic Foundations of Robotics V*, pages 541–558. Springer, 2004. [1.2.1](#)
- A. Gasparri, G. Oliva, and S. Panzieri. Path planning using a lazy spatial network PRM. In *Control and Automation. MED. 17th Mediterranean Conference on*, pages 940–945. IEEE, 2009. [2.3.1](#)
- Russell Gayle, Stephane Redon, Avneesh Sud, Ming C Lin, and Dinesh Manocha. Efficient motion planning of highly articulated chains using physics-based sampling. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 3319–3326. IEEE, 2007. [1.2.1](#)
- M. Gentilucci, U. Castiello, ML Corradini, M. Scarpa, C. Umiltà, and G. Rizzolatti. Influence of different types of grasping on the transport component of prehension movements. *Neuropsychologia*, 29(5):361–378, 1991. ISSN 0028-3932. [2.3.2](#)
- M. Gentilucci, S. Chieffi, M. Scarpa, and U. Castiello. Temporal coupling between transport and grasp components during prehension movements: effects of visual perturbation. *Behavioural Brain Research*, 47(1):71–82, 1992. ISSN 0166-4328. [2.2](#)
- M. Gienger, M. Toussaint, and C. Goerick. Task maps in humanoid robot manipulation. In *Intelligent Robots and Systems. IROS. IEEE/RSJ International Conference on*, pages 2758–2764. IEEE, 2008. [2.3.1](#)
- Michael Gienger, H Janben, and Christian Goerick. Exploiting task intervals for whole body robot control. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 2484–2490. IEEE, 2006. [5.2](#)
- E. Gribovskaya and A. Billard. Learning nonlinear multi-variate motion dynamics for real-time position and orientation control of robotic manipulators. In *Proceedings of IEEE-RAS International Conference on Humanoid Robots*, pages 472–477. IEEE, 2009. [2.2](#), [2.3.3](#), [2.4.1](#), [2.4.3](#), [2.4.3](#)

- E Gribovskaya, SM Khansari Zadeh, and A Billard. Learning non-linear multivariate dynamics of motion in robotic manipulators. *International Journal of Robotics Research*, 30(1):80–117, 2011. [1.2.2](#)
- D.B. Grimes, D.R. Rashid, and R.P.N. Rao. Learning nonparametric models for probabilistic imitation. *Advances in Neural Information Processing Systems*, 19:521, 2007. ISSN 1049-5258. [2.2](#)
- Daniel H Grollman and Aude Billard. Donut as I do: Learning from failed demonstrations. In *International Conference on Robotics and Automation*, Shanghai, May 2011. IEEE. [2.3.3](#)
- Joachim Gulke, Nikolaus J. Wachter, Thomas Geyer, Hendrik Schödl, Goran Apic, and Martin Mentzel. Motion coordination patterns during cylinder grip analyzed with a sensor glove. *The Journal of Hand Surgery*, 35(5):797 – 806, 2010. ISSN 0363-5023. doi: DOI:10.1016/j.jhsa.2009.12.031. [2.3.2](#)
- P. Haggard and A. Wing. Coordinated responses following mechanical perturbation of the arm during prehension. *Experimental Brain Research*, 102(3): 483–494, 1995. ISSN 0014-4819. [2.3.2](#)
- K. Harada, K. Kaneko, and F. Kanehiro. Fast grasp planning for hand/arm systems based on convex model. Robotics and Automation. ICRA. IEEE International Conference on, 2008. [2.3.1](#)
- Micha Hersch, Florent Guenter, Sylvain Calinon, and Aude Billard. Dynamical system modulation for robot learning via kinesthetic demonstrations. *Robotics, IEEE Transactions on*, 24(6):1463–1467, 2008. [1.2.2](#)
- B. Hoff and M.A. Arbib. Models of trajectory formation and temporal interaction of reach and grasp. *Journal of Motor Behavior*, 25(3):175–192, 1993. ISSN 0022-2895. [2.3.2](#)
- Heiko Hoffmann. Target switching in curved human arm movements is predicted by changing a single control parameter. *Experimental brain research*, 208(1): 73–87, 2011. [1.2.2](#), [3.1](#)
- Kaijen Hsiao, Paul Nangeroni, Manfred Huber, Ashutosh Saxena, and Andrew Y. Ng. Reactive grasping using optical proximity sensors. In *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, pages 2098–2105. Institute of Electrical and Electronics Engineers, May 2009. doi: 10.1109/ROBOT.2009.5152849. [2.3.1](#)
- J.H. Hwang, R.C. Arkin, and D.S. Kwon. Mobile robots at your fingertip: Bezier curve on-line trajectory generation for supervisory control. In *Intelligent Robots and Systems (IROS). Proceedings. IEEE/RSJ International Conference on*, volume 2, pages 1444–1449. IEEE, 2003. ISBN 0780378601. [2.3.3](#)
- A.J. Ijspeert, J. Nakanishi, and S. Schaal. Movement imitation with nonlinear dynamical systems in humanoid robots. In *Robotics and Automation. Proceedings. ICRA. IEEE International Conference on*, volume 2, pages 1398–1403. IEEE, 2002. ISBN 0780372727. [2.3.3](#)
- Auke Jan Ijspeert, Jun Nakanishi, and Stefan Schaal. Trajectory formation for imitation with nonlinear dynamical systems. In *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, volume 2, pages 752–757. IEEE, 2001. [1.2.2](#)

- Tetsunari Inamura, Iwaki Toshima, and Yoshihiko Nakamura. Acquiring motion elements for bidirectional computation of motion recognition and generation. In *Experimental robotics viii*, pages 372–381. Springer, 2003. [1.2.2](#)
- H. Jaeger, M. Lukosevicius, D. Popovici, and U. Siewert. Optimization and applications of echo state networks with leaky-integrator neurons. *Neural Networks*, 20(3):335–352, 2007. [3.2](#)
- Léonard Jaillet, Juan Cortés, and Thierry Siméon. Sampling-based path planning on configuration-space costmaps. *Robotics, IEEE Transactions on*, 26(4):635–646, 2010. [5.3](#)
- LS Jakobson and MA Goodale. Factors affecting higher-order movement planning: a kinematic analysis of human prehension. *Experimental Brain Research*, 86(1):199–208, 1991. ISSN 0014-4819. [2.2](#)
- M. Jeannerod. The timing of natural prehension movements. *Journal of Motor Behavior*, 16(3):235, 1984. [2.2](#), [2.3.2](#)
- Lydia Kavraki, Petr Svestka, Jean claude Latombe, and Mark Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. In *IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION*, pages 566–580, 1996a. [1.2.1](#), [1.2.1](#)
- Lydia E Kavraki, Jean-Claude Latombe, Rajeev Motwani, and Prabhakar Raghavan. Randomized query processing in robot path planning. In *Proceedings of the twenty-seventh annual ACM symposium on Theory of computing*, pages 353–362. ACM, 1995. [1.2.1](#)
- Lydia E Kavraki, Petr Svestka, J-C Latombe, and Mark H Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *Robotics and Automation, IEEE Transactions on*, 12(4):566–580, 1996b. [5.3](#)
- M. Kawato. Internal models for motor control and trajectory planning. *Current opinion in neurobiology*, 9(6):718–727, 1999. [2.2](#)
- Mohammad Keshmiri, Mehdi Keshmiri, and Abolfazl Mohebbi. Augmented online point to point trajectory planning, a new approach in catching a moving object by a manipulator. In *Control and Automation (ICCA), 8th IEEE International Conference on*, pages 1349–1354. IEEE, June 2010. doi: 10.1109/ICCA.2010.5524431. [2.3.3](#)
- S. M. Khansari Zadeh and A. Billard. Bm: An iterative method to learn stable non-linear dynamical systems with gaussian mixture models, 2010a. [1.2.2](#)
- S. M. Khansari Zadeh and Aude Billard. Learning Stable Non-Linear Dynamical Systems with Gaussian Mixture Models. *IEEE Transaction on Robotics*, 2011a. [3.1](#), [3.2](#), [3.5](#)
- Seyed Mohammad Khansari Zadeh and Aude Billard. Learning stable nonlinear dynamical systems with gaussian mixture models. *Robotics, IEEE Transactions on*, 27(5):943–957, 2011b. [1.2.2](#)
- S.M. Khansari Zadeh and A. Billard. Imitation learning of globally stable non-linear point-to-point robot motions using nonlinear programming. In *Intelligent Robots and Systems (IROS), IEEE/RSJ International Conference on*, pages 2676–2683. IEEE, 2010b. [2.2](#), [2.3.3](#), [2.4.1](#), [2.4.3](#), [A.1](#)

- Oussama Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *The international journal of robotics research*, 5(1):90–98, 1986. [1.2.1](#)
- Pradeep Khosla and Richard Volpe. Superquadric artificial potentials for obstacle avoidance and approach. In *Robotics and Automation, 1988. Proceedings., 1988 IEEE International Conference on*, pages 1778–1784. IEEE, 1988. [1.2.1](#)
- S. Kim, A. Shukla, and A. Billard. Catching objects in flight. *Robotics. IEEE Transactions on*, 2014.
- Seungsu Kim, Elena Gribovskaya, and Aude Billard. Learning Motion Dynamics to Catch a Moving Object. In *10th IEEE-RAS International Conference on Humanoid Robots*. IEEE, 2010. [2.5.3](#), [5](#)
- Ross A Knepper and Matthew T Mason. Real-time informed path sampling for motion planning search. *The International Journal of Robotics Research*, 31(11):1231–1250, 2012. [5.3](#)
- Daniel E Koditschek. Exact robot navigation by means of potential functions: Some topological considerations. In *Robotics and Automation. Proceedings. 1987 IEEE International Conference on*, volume 4, pages 1–6. IEEE, 1987. [1.2.1](#)
- Daniel E Koditschek. Robot planning and control via potential functions. In *The robotics review 1*, pages 349–367. MIT Press, 1989. [1.2.1](#)
- D. Kragic, A.T. Miller, and P.K. Allen. Real-time tracking meets online grasp planning. In *Robotics and Automation. Proceedings. ICRA. IEEE International Conference on*, volume 3, pages 2460–2465. IEEE, 2001. doi: 10.1109/ROBOT.2001.932992. [2.3.1](#)
- James J Kuffner and Steven M LaValle. Rrt-connect: An efficient approach to single-query path planning. In *Robotics and Automation, 2000. Proceedings. ICRA '00. IEEE International Conference on*, volume 2, pages 995–1001. IEEE, 2000. [1.2.1](#), [1.2.1](#), [5.2](#)
- Jean-Claude Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, 1991. [1.2.1](#)
- S. LaValle and J. Kuffner. Rapidly-exploring random trees: Progress and prospects. In *Algorithmic and computational robotics: new directions: the fourth Workshop on the Algorithmic Foundations of Robotics*, page 293. AK Peters, Ltd., 2001. ISBN 156881125X. [2.3.1](#)
- ETY Lee. Comments on some b-spline algorithms. *Computing*, 36(3):229–238, 1986. [1.2.2](#)
- J. Lee. Dynamic gradient approaches to compute the closest unstable equilibrium point for stability region estimate and their computational limitations. *Automatic Control, IEEE Transactions on*, 48(2):321–324, 2003. [3.3](#)
- Andre Lemme, K Neumann, RF Reinhart, and Jochen J Steil. Neural learning of vector fields for encoding stable dynamical systems. *Neurocomputing*, 141: 3–14, 2014. [4.2](#)
- Luka Lukic, José Santos-Victor, and Aude Billard. Learning robotic eye–arm–hand coordination from human demonstration: a coupled dynamical systems approach. *Biological cybernetics*, 108(2):223–248, 2014. [6.1.2](#)

- Mantas LukosEvičius and Herbert Jaeger. Reservoir computing approaches to recurrent neural network training. *Computer Science Review*, 3(3):127–149, 2009. [1.2.2](#)
- I Macêdo and R Castro. Learning divergence-free and curl-free vector fields with matrix-valued kernels. Technical report, Instituto Nacional de Matematica Pura e Aplicada, 2008. [4.2](#)
- Ives Macêdo, João Paulo Gois, and Luiz Velho. Hermite interpolation of implicit surfaces with radial basis functions. In *Computer Graphics and Image Processing (SIBGRAPI), 2009 XXII Brazilian Symposium on*, pages 1–8. IEEE, 2009. [4.2](#)
- Jim Mainprice, Emrah Akin Sisbot, Léonard Jaillet, Juan Cortés, Rachid Alami, and Thierry Siméon. Planning human-aware motions using a sampling-based costmap planner. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 5012–5017. IEEE, 2011. [5.3](#)
- R.G.J. Meulenbroek, D.A. Rosenbaum, C. Jansen, J. Vaughan, and S. Vogt. Multijoint grasping movements. *Experimental Brain Research*, 138(2):219–234, 2001. ISSN 0014-4819. [2.3.2](#)
- Charles A Micchelli and Massimiliano Pontil. On learning vector-valued functions. *Neural Computation*, 17(1):177–204, 2005. [4.2](#)
- A.N. Michel and J.A. Farrell. Associative memories via artificial neural networks. *Control Systems Magazine, IEEE*, 10(3):6–17, apr 1990. ISSN 0272-1708. doi: 10.1109/37.55118. [3.2](#)
- A.T. Miller, S. Knoop, H.I. Christensen, and P.K. Allen. Automatic grasp planning using shape primitives. In *Robotics and Automation. Proceedings. ICRA. IEEE International Conference on*, volume 2, pages 1824–1829. IEEE, 2003. ISBN 0780377362. [2.2](#)
- A.R. Mitz, M. Godschalk, and S.P. Wise. Learning-dependent neuronal activity in the premotor cortex: activity during the acquisition of conditional motor associations. *Journal of Neuroscience*, 11(6):1855, 1991. [2.3.2](#)
- Hiroyuki Miyamoto, Stefan Schaal, Francesca Gandolfo, Hiroaki Gomi, Yasuhiro Koike, Rieko Osu, Eri Nakano, Yasuhiro Wada, and Mitsuo Kawato. A kendama learning robot based on bi-directional theory. *Neural networks*, 9(8):1281–1302, 1996. [1.2.2](#)
- S Mohammad Khansari Zadeh and Aude Billard. Learning control lyapunov function to ensure stability of dynamical system-based robot reaching motions. *Robotics and Autonomous Systems*, 62(6):752–765, 2014. [1.2.2](#)
- Andrew Moore. Acquisition of dynamic control knowledge for a robotic manipulator. In Michael B. Morgan, editor, *Proceedings of the 7th International Conference on Machine Learning*, pages 244–252, 340 Pine Street, 6th Fl., San Francisco, CA 94104, 1990. Morgan Kaufmann. [1.2.2](#)
- A. Morales, T. Asfour, P. Azad, S. Knoop, and R. Dillmann. Integrated grasp planning and visual object localization for a humanoid robot with five-fingered hands. In *Intelligent Robots and Systems. IEEE/RSJ International Conference on*, pages 5663–5668. IEEE, 2007. ISBN 1424402581. [2.3.1](#)

- S Muench, J Kreuziger, M Kaiser, and R Dillman. Robot programming by demonstration (rpd)-using machine learning and user interaction methods for the development of easy and comfortable robot programming systems. In *Proceedings of the International Symposium on Industrial Robots*, volume 25, pages 685–685. INTERNATIONAL FEDERATION OF ROBOTICS, & ROBOTIC INDUSTRIES, 1994. [1.2.2](#)
- Roderick Murray-Smith, Tor A Johansen, and Robert Shorten. On transient dynamics, off-equilibrium behaviour and identification in blended multiple model structures. In *European control conference*, pages BA–14. Springer, 1999. [4.2](#)
- Chrystopher L Nehaniv and Kerstin Dautenhahn. Of hummingbirds and helicopters: An algebraic framework for interdisciplinary studies of imitation and its applications. In *Interdisciplinary Approaches to Robot Learning*, 1999. [1.2.2](#)
- Klaus Neumann, Andre Lemme, and Jochen J Steil. Neural learning of stable dynamical systems based on data-driven lyapunov candidates. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 1216–1222. IEEE, 2013. [1.2.2](#), [4.4.3](#)
- A. O’Hagan. Some bayesian numerical analysis. *Bayesian Statistics*, 4:345–363, 1992. [4.2](#)
- E. Oztop and M.A. Arbib. Schema design and implementation of the grasp-related mirror neuron system. *Biological cybernetics*, 87(2):116–140, 2002. [2.3.2](#)
- P. Pastor, H. Hoffmann, and Stefan Schaal. Movement generation by learning from demonstration and generalization to new targets. In *Adaptive Motion of Animals and Machines (AMAM)*, 2008. [2.2](#)
- Peter Pastor, Heiko Hoffmann, Tamim Asfour, and Stefan Schaal. Learning and generalization of motor skills by learning from demonstration. In *Robotics and Automation, 2009. ICRA ’09. IEEE International Conference on*, pages 763–768, may 2009. doi: 10.1109/ROBOT.2009.5152385. [3.1](#), [3.2](#)
- Y. Paulignan, M. Jeannerod, C. MacKenzie, and R. Marteniuk. Selective perturbation of visual input during prehension movements. *Experimental Brain Research*, 87(2):407–420, 1991. ISSN 0014-4819. [2.2](#)
- Y. Paulignan, C. MacKenzie, R. Marteniuk, and M. Jeannerod. The coupling of arm and finger movements during prehension. *Experimental Brain Research*, 79:431–435, 1994. ISSN 0014-4819. [2.3.2](#)
- John C. Platt. Sequential minimal optimization: A fast algorithm for training support vector machines. Technical report, Advances in Kernel Methods - Support Vector Learning, 1998. [1.3](#), [4.2](#), [4.3.3](#)
- Danil V Prokhorov and Lee A Feldkamp. Application of svm to lyapunov function approximation. In *Neural Networks, 1999. IJCNN’99. International Joint Conference on*, volume 1, pages 383–387. IEEE, 1999. [4.4.3](#)
- C. Rasmussen. Gaussian processes in machine learning. *Advanced Lectures on Machine Learning*, pages 63–71, 2004. [3.2](#)

- H. Reimann, I. Iossifidis, and G. Schöner. Autonomous movement generation for manipulators with multiple simultaneous constraints using the attractor dynamics approach. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 5470–5477. IEEE, 2011. [3.1](#)
- René Felix Reinhart and Jochen Jakob Steil. Neural learning and dynamical selection of redundant solutions for inverse kinematic control. In *Humanoid Robots (Humanoids), 2011 11th IEEE-RAS International Conference on*, pages 564–569. IEEE, 2011. [1.2.2](#), [4.2](#)
- Elon Rimon and Daniel E Koditschek. Exact robot navigation using artificial potential functions. *Robotics and Automation, IEEE Transactions on*, 8(5): 501–518, 1992. [1.2.1](#)
- Lorenzo Rosasco, Matteo Santoro, Sofia Mosci, Alessandro Verri, and Silvia Villa. A regularization approach to nonlinear variable selection. In *International Conference on Artificial Intelligence and Statistics*, pages 653–660, 2010. [4.2](#)
- M. Saling, S. Mescheriakov, E. Molokanova, GE Stelmach, and M. Berger. Grip reorganization during wrist transport: the influence of an altered aperture. *Experimental Brain Research*, 108(3):493–500, 1996. ISSN 0014-4819. [2.3.2](#)
- Joe Saunders, Chrystopher L Nehaniv, and Kerstin Dautenhahn. Teaching robots by moulding behavior and scaffolding the environment. In *Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*, pages 118–125. ACM, 2006. [1.2.2](#)
- J-P. Saut, A. Sahbani, S. El-Khoury, and V. Perdereau. Dexterous manipulation planning using probabilistic roadmaps in continuous grasp subspaces. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2907–2912. IEEE, 2007. [2.3.1](#)
- S. Schaal and C.G. Atkeson. Constructive incremental learning from only local information. *Neural Computation*, 10(8):2047–2084, 1998. ISSN 0899-7667. [2.3.3](#)
- S. Schaal, S. Kotosaka, and D. Sternad. Nonlinear dynamical systems as movement primitives. In *Humanoids. First IEEE-RAS International Conference on Humanoid Robots*. CD-Proceedings, 2000. [2.3.3](#)
- S. Schaal, C.G. Atkeson, and S. Vijayakumar. Scalable techniques from non-parametric statistics for real time robot learning. *Applied Intelligence*, 17(1): 49–60, 2002. [3.2](#)
- S. Schaal, A. Ijspeert, and A. Billard. Computational approaches to motor learning by imitation. *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences*, 358(1431):537, 2003. ISSN 0962-8436. [1.2.2](#)
- B. Schölkopf and A.J. Smola. *Learning with kernels: Support vector machines, regularization, optimization, and beyond*. MIT press, 2001. [3.4.1](#)
- Bernhard Schölkopf, Alex J Smola, Robert C Williamson, and Peter L Bartlett. New support vector algorithms. *Neural computation*, 12(5):1207–1245, 2000. [4.2](#)

- G. Schöner and M. Dose. A dynamical systems approach to task-level system integration used to plan and control autonomous vehicle motion. *Robotics and Autonomous Systems*, 10(4):253–267, 1992. [3.1](#)
- G. Schöner, M. Dose, and C. Engels. Dynamics of behavior: Theory and applications for autonomous robot architectures. *Robotics and Autonomous Systems*, 16(2):213–245, 1995. [3.1](#)
- Gursel Serpen. Empirical approximation for lyapunov functions with artificial neural nets. In *IEEE International Joint Conference on Neural Networks (IJCNN)*, volume 2, pages 735–740. IEEE, 2005. [4.4.3](#)
- Alexander Shkolnik and Russ Tedrake. Path planning in 1000+ dimensions using a task-space voronoi bias. In *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, pages 2061–2067. IEEE, 2009. [5.6](#)
- A. Shukla and A. Billard. Coupled dynamical system based arm-hand grasping model for learning fast adaptation strategies under real-time perturbations. In *Proceedings of Robotics: Science and Systems VII*, pages 313 – 320. MIT Press, 2011. ISBN 978-0-262-51779-9.
- A. Shukla and A. Billard. Coupled dynamical system based arm-hand grasping model for learning fast adaptation strategies. *Robotics and Autonomous Systems*, 60(3):424 – 440, 2012a. ISSN 0921-8890. doi: 10.1016/j.robot.2011.07.023.
- A. Shukla and A. Billard. Augmented-SVM: Automatic space partitioning for combining multiple non-linear dynamics. In *Neural Information Processing Systems (NIPS) 25*, pages 1025–1033, 2012b. [4.2](#), [4.4.3](#)
- A. Shukla and A. Billard. Augmented-svm for gradient observations with application to learning multiple-attractor dynamics. In *Support Vector Machines Applications, Chapter 1*, pages 1–21. Springer International Publishing, 2014. ISBN 978-3-319-02299-4. doi: 10.1007/978-3-319-02300-7_1.
- Ercan Solak, Roderick Murray-Smith, William E Leithead, Douglas J Leith, and Carl Edward Rasmussen. Derivative observations in gaussian process models of dynamic systems. In *Advances in Neural Information Processing Systems (NIPS) 14*. MIT Press, 2003. [4.2](#)
- Russ Tedrake, Ian R Manchester, Mark Tobenkin, and John W Roberts. Lqr-trees: Feedback motion planning via sums-of-squares verification. *The International Journal of Robotics Research*, 2010. [1.2.1](#)
- Howell Tong. *Non-linear time series: a dynamical system approach*. Oxford University Press, 1990. [1.2.2](#)
- M. Toussaint. Robot trajectory optimization using approximate inference. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1049–1056. ACM, 2009. [1.2.1](#), [5.6](#)
- M. Toussaint and C. Goerick. A bayesian view on motor control and planning. *From Motor Learning to Interaction Learning in Robots*, pages 227–252, 2010. [5.6](#)
- Ales Ude, Andrej Gams, Tamim Asfour, and Jun Morimoto. Task-specific generalization of discrete and periodic dynamic movement primitives. *Robotics, IEEE Transactions on*, 26(5):800–815, 2010. [1.2.2](#)

- A. Ulloa and D. Bullock. A neural circuit for coordinating reaching with grasping: autocompensating variable initial apertures, perturbations to target size, and perturbations to target orientation. In *Neural Networks. Proceedings. IJCNN. International Joint Conference on*, volume 2, pages 1047–1052. IEEE, 2002. ISBN 0780370449. [2.3.2](#)
- Antonio Ulloa and Daniel Bullock. A neural network simulating human reach-grasp coordination by continuous updating of vector positioning commands. *Neural Networks*, 16(8):1141 – 1160, 2003. ISSN 0893-6080. doi: DOI:10.1016/S0893-6080(03)00079-0. [2.3.2](#)
- Chris Urmson and Reid G Simmons. Approaches for heuristically biasing rrt growth. In *IROS*, pages 1178–1183, 2003. [5.3](#), [5.5](#)
- Niko Vahrenkamp, Dmitry Berenson, Tamim Asfour, James Kuffner, and Rüdiger Dillmann. Humanoid motion planning for dual-arm manipulation and re-grasping tasks. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 2464–2470. IEEE, 2009. [2.3.1](#), [5.6](#), [5.6](#)
- JM Vilaplana, JF Batlle, and JL Coronado. A neural model of hand grip formation during reach to grasp. In *Systems, Man and Cybernetics, IEEE International Conference on*, volume 1, pages 542–546. IEEE, 2005. ISBN 0780385667. [2.3.2](#)
- Jack Wang, Aaron Hertzmann, and David M Blei. Gaussian process dynamical models. In *Advances in neural information processing systems*, pages 1441–1448, 2005. [1.2.2](#)
- Daniel M Wolpert and Mitsuo Kawato. Multiple paired forward and inverse models for motor control. *Neural Networks*, 11(7):1317–1329, 1998. [1.2.2](#)
- Katsu Yamane, James J Kuffner, and Jessica K Hodgins. Synthesizing animations of human manipulation tasks. In *ACM Transactions on Graphics (TOG)*, volume 23, pages 532–539. ACM, 2004. [1.2.2](#)
- Gu Ye and Ron Alterovitz. Demonstration-guided motion planning. In *International Symposium on Robotics Research (ISRR)*, 2011. [5.3](#)

Ashwini Shukla

DOB: 24th June 1985
Ph.D Student - IV Year

Rue de la Blancherie 5
Chavannes, Switzerland - 1022
T: +41-78-693-6657
E: ashwini.shukla@epfl.ch

Education

Ecole Polytechnique de Federale de Lausanne (EPFL), Switzerland

Ph.D, Machine Learning and Robotics — 2010-present (CGPA – 5.5 / 6.0)

- Conducted research on motor skill learning from human demonstrations using SVM and Dynamical Systems.
- Teaching Assistant for the “Applied Machine Learning” course. Conceived and supervised two semester projects and one summer internship.
- Timely deliverables, reports and demonstrations to project reviewers of two EU funded projects “First-MM” and “AlterEgo”.

Indian Institute of Technology Kanpur (IIT-K), India

M.Tech, Mechanical Engineering — 2008-2009 (CGPA – 10.0 / 10.0)

- Applied non-linear optimization to solve path planning problems for manipulators.
- Integrated as a C++ module in the robot path planning framework YANTRIKA®.

Indian Institute of Technology Kanpur (IIT-K), India

B.Tech, Mechanical Engineering — 2004-2008 (CGPA – 8.2 / 10.0)

- Developed skills in Theory of Optimization, JAVA/C++ Programming and Robotics.
- Served as the Institute Robotics Club Coordinator for 2007-2008.
- Represented IIT Kanpur at ROBOCON 2006 as part of a 6 membered team.

Work

Deloitte Consulting India

Business Technology Analyst — 2009-2010

- Worked as a JAVA developer in a team of 9 translating client requirements to software.
- Trained on aspects of technological consulting and software development lifecycle.
- Received technical training on JAVA-Struts, Spring and Hibernate frameworks.

University West / Volvo Aero

Summer Intern — May-July 2007

- Developed path-planning algorithms for rapid prototyping of aircraft parts.
- Programmed the planner as a VC++ module and interfaced with ROBCAD simulation.

Courses

Technical

- Estimation theory, Dynamical systems for Engineers, Optimal Control

Software

- Sun Certified JAVA Programmer, Algorithms and Data Structures, NLP, IR.

Languages

- TOEFL (score 115/120), French (level A0).

Skills

Machine Learning

- SVM, Max-Ent models, Bayesian regression/classification, Gaussian Processes, Gaussian Mixture Models.

Programming

- JAVA (SCJP certified), C/C++ (Proficient), MATLAB (Proficient), Python (Beginner), SQL

Tools

- Eclipse, CMake, Version Control via SVN, BZR.

Operating Systems

- Linux (Ubuntu) including administrative tasks, WINDOWS® and MACINTOSH®

Key Achievements

- Ranked 165 (**top 5%**) at codeeval.com for solving programming puzzles.
- All India Rank 804 (**99.73 percentile**) and 206 (**99.97 percentile**) in the all-India engineering entrance examinations IIT-JEE and AIEEE respectively.
- Awarded for holding **Rank-1** in the postgraduate class.
- Received Academic Excellence Award for being among the **top 7%** students in 2008-2009 semester postgraduate class.

Conference Publications & Talks

- **Swiss Machine Learning Day 2013.**
Support Vector Algorithms for Gradient Observations.
- **Neural Information Processing Systems (NIPS) 2012.**
Augmented-SVM: Automatic space partitioning for combining multiple non-linear dynamics.
- **International Symposium on System Integration (SII) 2012.**
Reaching and grasping kitchenware objects.
- **Robotics: Science and Systems (RSS) 2011.**
Coupled dynamical system based hand-arm grasp planning under real time-perturbations.

Journal & Chapter Publications

- **Support Vector Machine Applications. Springer, In Press, Year 2013.**
Augmented-SVM for gradient observations with application to learning multi-attractor dynamics.
- **Robotics and Autonomous Systems. Vol. 60, Pages 424 - 440, Year 2012.**
Coupled dynamical system based arm-hand grasping model for learning fast adaptation.
- **Robotics and Autonomous Systems. Vol. 61, Pages 209 - 220, Year 2012.**
A Direct Variational Method for Planning Monotonically Optimal Paths for Manipulators.

Personal Information

- 28 years old, married (no children), Indian citizen, Swiss residence permit B.
- **Hobbies:** Skiing, swimming, cricket and playing guitar.